NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

OBJECT LOCALIZATION AND RANGING USING STEREO VISION FOR USE ON AUTONOMOUS GROUND VEHICLES

by

Keith Andrew Baravik

June 2009

Thesis Advisor:	Richard Harkins
Co-Advisor:	Nancy Haegel

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2009	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Object Localization and Ranging Using Stereo Vision for Use on Autonomous Ground Vehicles		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Keith Andrew Baravik			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

This thesis integrates stereo-vision into existing NPS robot architecture. It demonstrates that image cross correlation can be used to measure ranges as theory predicts. It also demonstrates that objects can be ranged and stored into a database map for later use as common reference points in position determination.

Small Unmanned Ground Vehicles (UGV), developed using commercial-off-the-shelf (COTS) technologies are of particular interest for this robotic vision application. To perform their designated missions, these devices require accurate position information. Most devices will determine that position using a Global Positioning System (GPS) receiver; however, the signal is vulnerable to jamming and becomes degraded when not provided a clear view of the sky. Similarly, the error in dead reckoning (DR) systems increases with time if not reset using a known reference. The fusion of stereo vision technology with GPS and DR systems is ideal for use in the design of a command and control module of an unmanned vehicle that is capable of operating autonomously in an environment where traditional position determination loses satellite signals or requires a known reference point to reset uncertainty in position.

14. SUBJECT TERMS Robotic Vision, Unmanned Ground Vehicle			**15. NUMBER OF PAGES** 85
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

OBJECT LOCALIZATIONAND RANGING USING STEREO VISION FOR USE ON AUTONOMOUS GROUND VEHICLES

Keith A. Baravik
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1997

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN PHYSICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2009

Author: Keith Andrew Baravik

Approved by: Richard Harkins
Thesis Advisor

Nancy Haegel
Co-Advisor

James Luscombe
Chairman, Department of Physics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis integrates stereo-vision into existing NPS robot architecture. It demonstrates that image cross correlation can be used to measure ranges as theory predicts. It also demonstrates that objects can be ranged and stored into a database map for later use as common reference points in position determination.

Small Unmanned Ground Vehicles (UGV), developed using commercial-off-the-shelf (COTS) technologies are of particular interest for this robotic vision application. To perform their designated missions, these devices require accurate position information. Most devices will determine that position using a Global Positioning System (GPS) receiver; however, the signal is vulnerable to jamming and becomes degraded when not provided a clear view of the sky. Similarly, the error in dead reckoning (DR) systems increases with time if not reset using a known reference. The fusion of stereo vision technology with GPS and DR systems is ideal for use in the design of a command and control module of an unmanned vehicle that is capable of operating autonomously in an environment where traditional position determination loses satellite signals or requires a known reference point to reset uncertainty in position.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

A special thanks to my wife and family who provided me with encouragement, support, and motivation to start work on my thesis — sooner rather than later.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The exponential growth of computing power, development of Micro Electro Mechanical Systems (MEMS), micro sensors, and improved battery capacity have made it possible to replace a person with a robotic solution for the performance of a task in a hazardous location. The military community is particularly interested in the concept of autonomy, since the demands placed on the operator of even a semi-autonomous robot reduce situational awareness and safety. In response, ongoing research in robotic technology has developed capabilities for Proximal Autonomy in which robots conduct missions in the vicinity of their human leaders; however, the integration of sensors and systems within these advanced platforms is still in development [1].

This thesis integrates a stereo vision system into the existing architecture of the Naval Postgraduate School's Applied Physics laboratory robot designs. Specifically, the system is evaluated for object localization and ranging. Accurate object localization and ranging is vital for developing autonomous site mapping and navigation routines.

A. MILITARY INVESTMENT IN MAN PORTABLE AUTONOMOUS ROBOTS

Recently, due to the nature of the conflicts in Iraq and Afghanistan, there has been significant investment in removing the human element from dangerous tasks such as Explosive Ordinance Disposal and scouting potentially hostile urban environments. Unmanned Ground Vehicles (UGV) are employed at an increased rate but remain mostly at the semi-autonomous level. The iRobot, PACBOT, and TALON robots are current examples. The sensors required for small UGVs to operate with full and proximal autonomy are readily available. While the current trend of robotics research is in the development of algorithms to integrate sensors for improved situational awareness, obstacle avoidance and position determination continue to be high priorities [2]. Specifically, if the robot cannot tell where it is or where it has been then it is impossible for it to intelligently determine where it is going in order to best compliment the needs of its operator.

B. ACTIVE RESEARCH WITHIN THE DoD

Global Positioning Systems (GPS) are good for determining position while Dead Reckoning (DR) can be accurate for systems under the right circumstances. UGVs designed to operate in urban environments can expect degraded GPS signals, which in turn affects the ability to DR, since a reliable reset position is not always available. Therefore, neither GPS nor DR is sufficient under these conditions.

Funded research in robotic vision is expected to fill the gap that occurs in DR systems when GPS is degraded to where it no longer provides a reference position to reduce position uncertainty. A search of the DoD Technical Information Center (DTIC) showed that funding is being provided for research into the topics of sensor resetting and normalization based on environment, ranging based on focus characteristics, advanced object identification algorithms, and many others at various academic institutions. Similarly, SPAWAR Systems Center is funding VisionRobotics Corp to research stereo vision scene mapping for Explosive Ordinance Disposal (EOD) Unmanned Ground Vehicles (UGV) [3]. Success in this area of research would provide a significant improvement in position mapping and object recognition for use in future autonomous applications.

II. CONCEPT OF FUNCTIONAL DESIGN

The concept of stereo vision in robotics is an extension of organic stereo vision that occurs naturally in most animals. The major components of the organic vision system are modeled with a mechanical device capable of performing a similar function. Evolution decided the placement and sensitivity of the organs used in organic vision, while the system designer of the robotic solution has greater flexibility in choosing components that best meet objectives.

A. ORGANIC MODEL OF VISION SYSTEMS

A simple model of an organic vision system includes two sensors, the eyes, connected to the brain by a tightly packed bundle of nerves. For this application, the design priority is to simplify the control and processing functions such that computation demands are within the capabilities of the onboard UGV processor. Therefore, in this model, the two eyes will remain stationary with respect to each other so that the acuity of the image is a function of: (1) the number of organic sensing cells, (2) field of view (FOV), and (3) distance to the object.

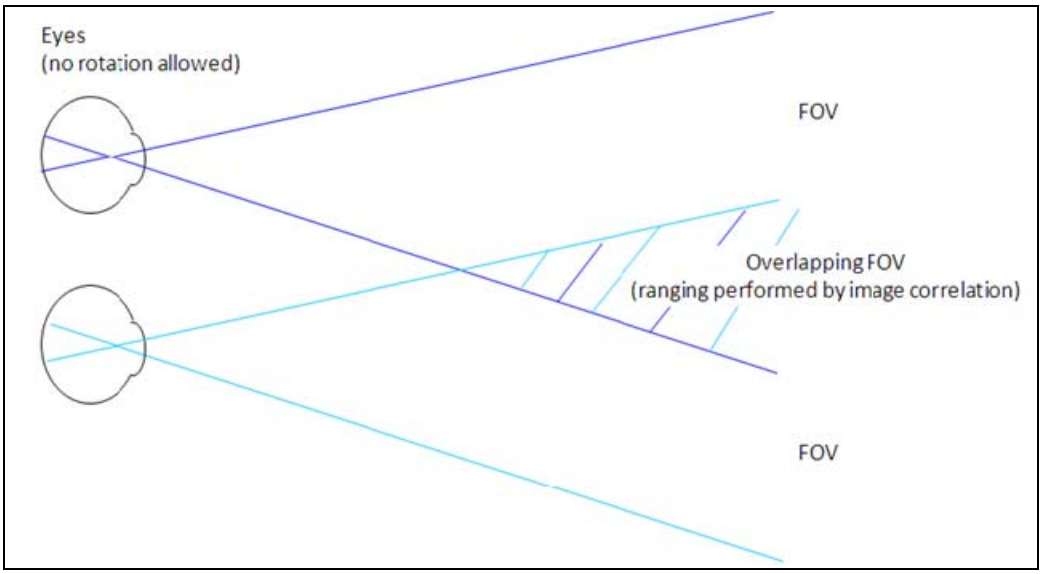


Figure 1. Diagram of simplified Organic Vision System

Figure 1 shows that image correlation can be performed in the area of FOV overlap.

1. Sensitivity of the Human Eye

The human eye varies from person to person based on an individual's unique body structure. To compare the acuity between differing sets of eyes, a standard, referred to as the Snellen Acuity, was adopted by eye care professionals. Under this standard, the average human eye has 20/20 vision with approximately the following characteristics:

- Concentration of cones on retina is $180,000\ mm^{-2}$ at center and decreases to less than $5000\ mm^{-2}$ as the image approaches the periphery of the field of view [4].
- Depth of the eye, from cornea to retina, is approximately 20 mm [5].

Similarly, the ability of the eye to focus and the FOV will be unique based on the individual. Because the concentration of cones in the eye decreases as the image extends outward, only a small FOV has a high degree of acuity. Research at the University of Texas Austin goes into much greater detail on this subject. It discusses how the information perceived using the eye is processed to develop a composite picture, however, the useful concept for this project is that only a small portion of a human's FOV is at high resolution [6].

Figure 2. Perceived variable resolution of fixed image (from [6])

The first image of Figure 1 shows that the human eye has a high resolution response under only a small FOV. The second image is a visual representation of points in the image that were focused on by a test subject to gather the full image at full resolution.

2. Calculated Acuity of the Human Eye

For the calculated resolution of the human eye, we will only consider concentration of cones on the retina since they are the receptors that distinguish color and most closely correlate to the pixels of a modern CMOS sensor. Figure 3 shows that the FOV for maximum resolution is approximately10-12 degrees. Outside of that FOV the resolution remains consistently low.

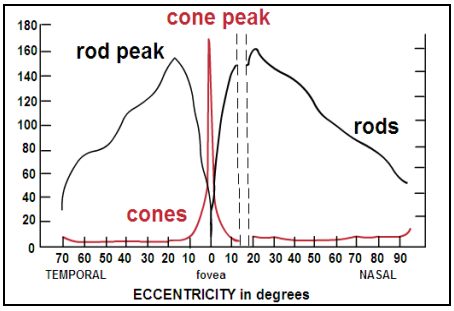


Figure 3. Relative Concentration of Cones in eye by FOV (from [7])

Therefore, given that the depth of the eye is approximately 20 mm and there are approximately two general areas of resolution, given by cone concentrations of $180,000$ cones mm^{-2} for high resolution and $5,000$ cones mm^{-2} for low resolution, see Figure 4, the actual resolution of the eye can be calculated from the following:

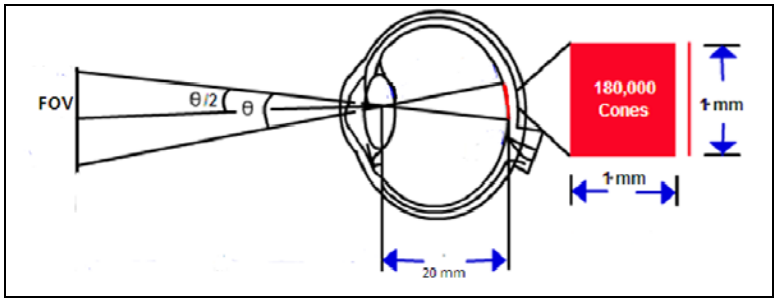


Figure 4. Diagram of optical acuity (from [7])

$$\#\text{ sensing areas } = \frac{C}{3}\pi r^2 = \frac{C}{3}\pi(20mm \cdot \tan\theta/_2)^2 \qquad (1.1)$$

Equation 1.1 assumes equal distribution of cones (C) over the area of FOV is divided by three to account for red, yellow, and green cones and is then sensed by the retina (πr^2).

It can then be calculated that for an 11 degree FOV, high resolution sight ($180,000$ cones mm^{-2}) will use approximately 0.7 million sensing areas, like pixels in a digital camera, and low resolution sight ($5,000$ cones mm^{-2}) will use 18,000 sensing areas.

B. MACHINE MODEL OF VISION SYSTEMS

A machine model of a vision system, Figure 5, is comparable to that of an organic vision system in that it has two primary sensors, transmissions lines from the sensors to the main processor, and a central processing unit to evaluate incoming data. The most important feature for this application is the sensitivity and acuity of the sensor employed in object identification and ranging. Greater sensitivity equates to a better ability to discern variations in color but comes at a cost of increased calculations per image. Similarly, increased acuity, more pixels per image, allows for better range resolution but at a cost of increased calculations when processing the images.

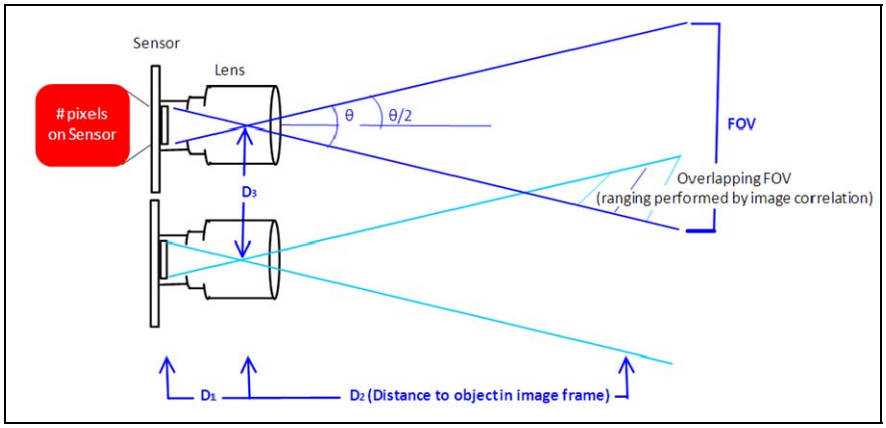


Figure 5. Diagram of Machine Vision System

1. Sensitivity and Acuity of Available CMOS Cameras

Advancements in fabrication technology has brought the acuity of commercially implemented sensors from 0.3 Megapixels in 1981 [8] to a height of 60 Megapixels that is scheduled for release in 2009 [9]. Similarly, sensitivity of image capture has improved as more capable onboard processing made it possible to perform real time calculations on images with higher bits per pixel ratios. Ideally, only the most sensitive and highest pixel

ratio sensors would be employed on the UGV in order to achieve the best range accuracy. In order to provide real time processing for a UGV, the sensor must not overwhelm the onboard processor with an abundance of information. The following sections will show that a reasonable level of accuracy can be obtained using sensors of relatively low resolution.

2. Calculation of Robotic Vision Acuity

Equation 1.1 calculated the number of sensing areas used by the eye for an image based on FOV, which is equivalent to the pixels stored in an image. For robotic vision, the number of pixels in the image is given by the characteristics of the installed sensor and the accuracy of the range calculation is determined using geometry.

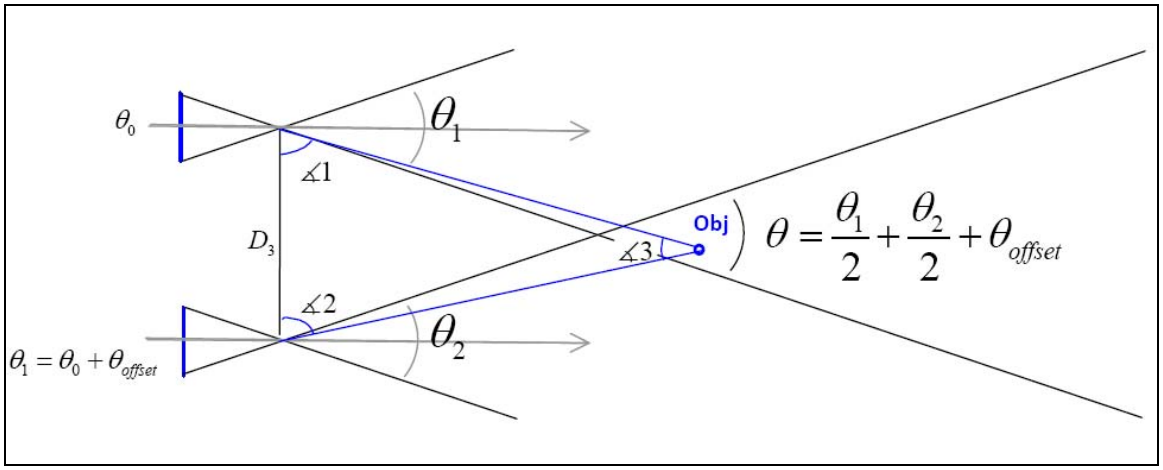


Figure 6. Diagram of angles used in Machine Vision range calculation

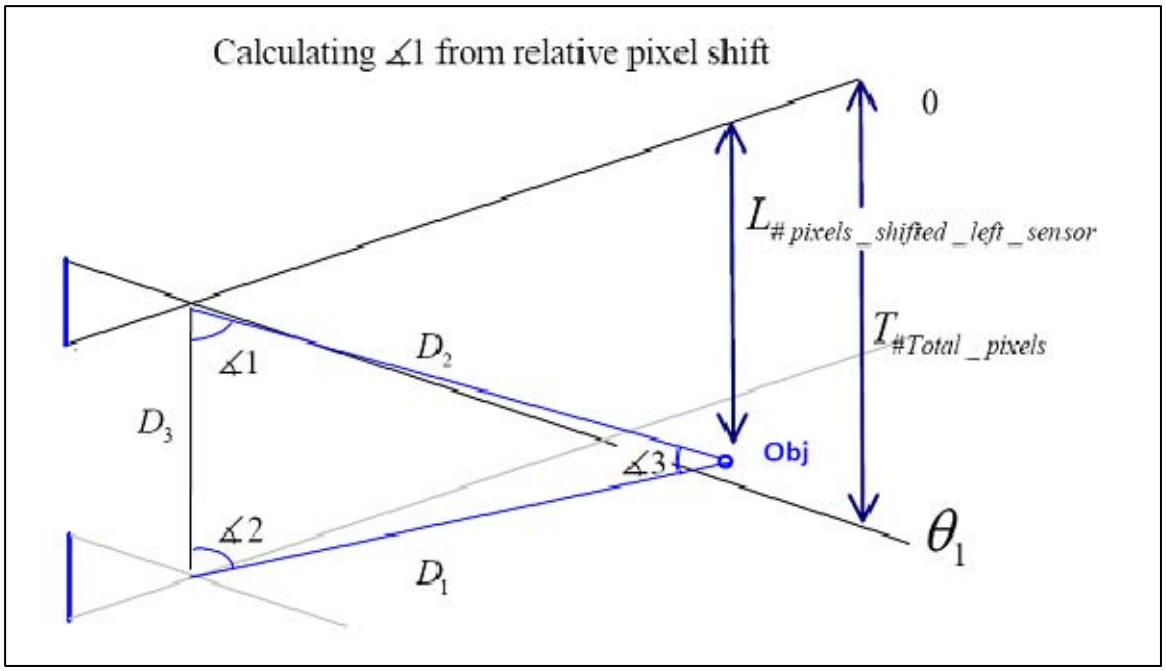


Figure 7. Diagram for calculation of interior angle ∠1 of range triangle

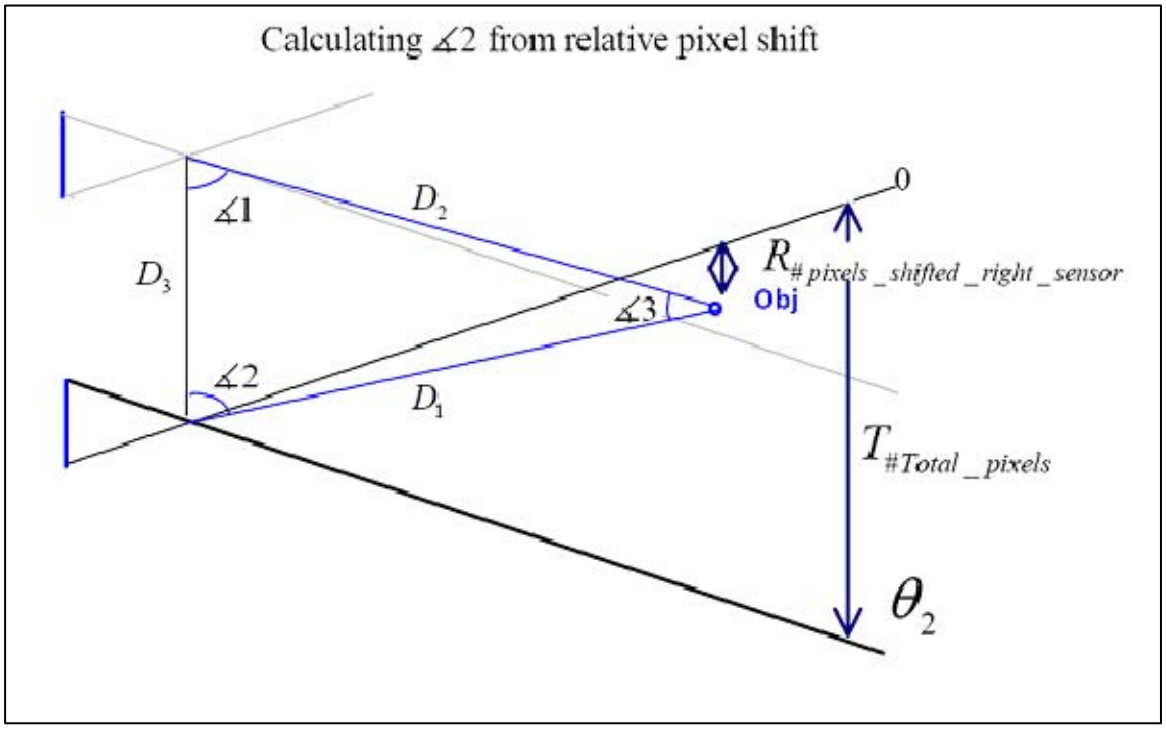


Figure 8. Diagram for calculation of interior angle ∠2 of range triangle

From the diagrams of Figures 6, 7, and 8, the following can be said about the interior angles that are made by the triangle formed from an object in the image plane and the two focal points of the installed cameras.

$$\measuredangle 1 = 90 - \frac{\theta_1}{2} + \theta_1(1 - \frac{L}{T}) = 90 + \theta_1(\frac{1}{2} - \frac{L}{T}) \quad (1.2)$$

$$\measuredangle 2 = 90 - \frac{\theta_2}{2} + \theta_2(\frac{R}{T}) = 90 + \theta_2(\frac{R}{T} - \frac{1}{2}) - \theta_{offset} \quad (1.3)$$

$$\measuredangle 3 = 180 - \measuredangle 1 - \measuredangle 2 = \theta_1(\frac{L}{T} - \frac{1}{2}) - \theta_2(\frac{R}{T} - \frac{1}{2}) \quad (1.4)$$

The variables L, for the left sensor, and R, for the right sensor, refer to the number of pixels that the object being ranged is from the leftmost side of the image frame. T is then the total number of horizontal pixels that appear in the image. From the ratio between L or R and T, the interior angles can be calculated.

Using the relationships from Equations 1.2, 1.3, and 1.4, the minimum range that the system can determine is,

$$R_{\min} = \frac{D_3}{2\tan(\theta/2)} \quad (1.5)$$

Then, looking more closely at the triangle formed by the object and the two focal points of the cameras and applying the Law of Sines, the range to the object can be calculated.

$$\mathrm{R}_{object} = D_2 \sin(\measuredangle 1) = D_1 \sin(\measuredangle 2) = D_3 \frac{\sin(\measuredangle 1)\sin(\measuredangle 2)}{\sin(\measuredangle 3)} \quad (1.6)$$

This range calculation takes into account many variables and can be applied in the case where the sensor is moved in order to lengthen the value of D_3 as is depicted in Figure 9.

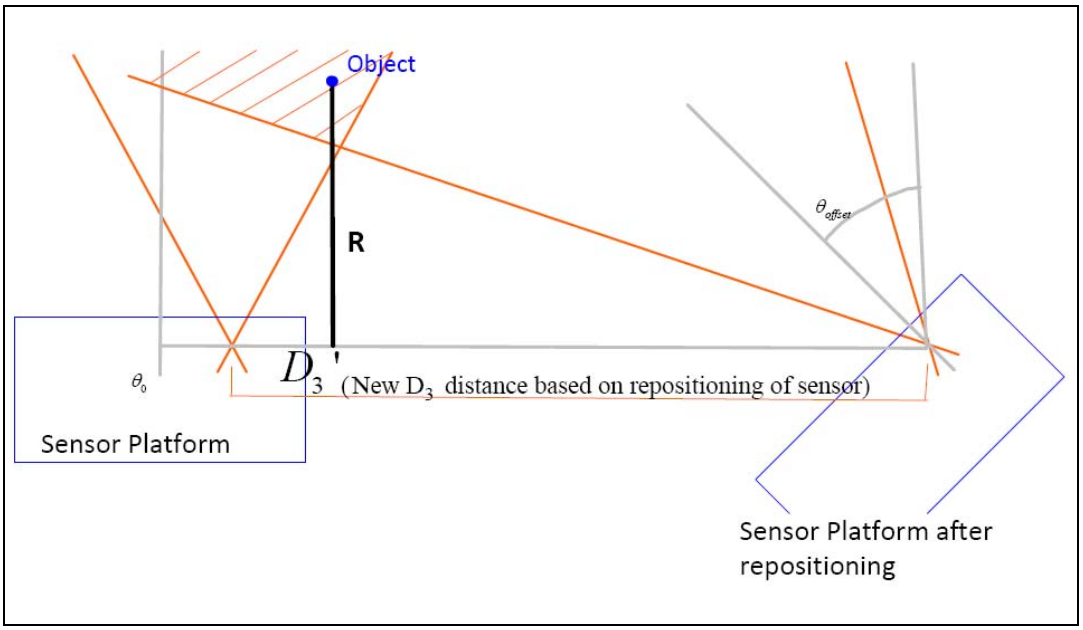


Figure 9. Diagram for range calculation after sensor repositioning.

Moving the sensor platform to lengthen the value of D_3 can significantly increase the utility of stereo vision by extending its effective range. However, you should also see in Figure 9, that repositioning the platform changes the point at which the calculated range is measured from. In some instances, as is shown in Figure 9, the reference point from which range is calculated does not even lie on the robot platform. Another issue that arises with the range calculation of Equation 1.6 after the sensor platform is repositioned, is that multiple calculations of sine functions must be made before a range is determined. Since there is inherent error in each of the angle and range measurements of Equation 1.6, the determined range will have four individual sources of error making it less reliable.

A solution to these problems is to simplify parameters so that the entire sensor is evaluated using similar triangle techniques. By doing this, you will see that the source of measurement error reduces to one parameter, actual range, with all other parameters remaining constant. It will also fix the point from which range is calculated to the same point on the robot at all times. This is shown in the next section as Equation 1.8. The constant of Equation 1.8 will be determined experimentally using a best fit of measured data in Chapter IV.

C. FURTHER SIMPLIFICATION OF STEREO VISION RANGE CALCUALTION

To simplify the control requirements of sensor operation, we eliminate the dynamic movements that normally accompany organic vision. To do this, two cameras are mounted on a circuit board which eliminates the variation between the orientation of the respective cameras and fixes the value of D_3. Similarly, fixed focal length lenses are used. Once the lenses are installed and adjusted such that both cameras use the same FOV, the range (R) to an object in the image plane is calculated using a scaled ratio of the total number of horizontal pixels (C) divided by the horizontal shift between the left image (L) and right image (R).

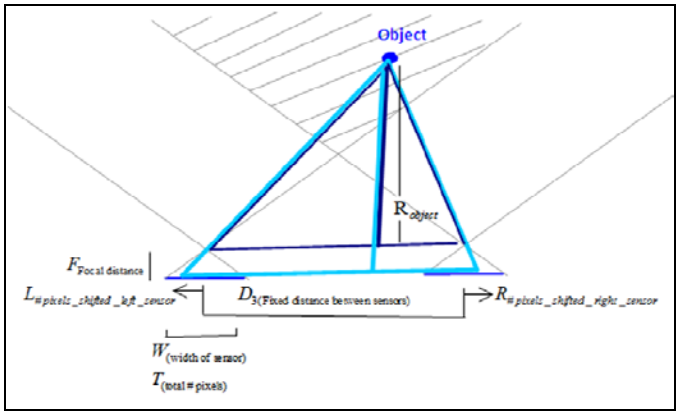


Figure 10. Range Calculation using Similar Triangles

$$\frac{R}{D_3} = \frac{R+F}{W(L/T - R/T)} \quad (1.7)$$

Solving for R and rearranging we have:

$$R = C_{(constant)} \frac{1}{L-R}, \quad \text{where } C = \frac{D_3 FT}{W} \quad (1.8)$$

The values for distance between sensors (D_3), focal length (F), total number of pixels across width of sensor (T), and width of sensor (W) that make up the constant (C) are depicted in Figure 10.

D. DATA PROCESSING TO OBTAIN POSITION INFORMATION USING MACHINE SYSTEM

Now that range has been determined mathematically, the problem becomes one of discerning individual objects from the recorded image frame from both cameras and matching the objects between images. This is where the greatest processing load is put on the UGV processor.

1. Object Recognition and Ranging

For object identification, two dimensional cross correlation must be performed for each object in one image with all the other objects in the other until a match is found or it can be determined that there is not a sufficient match. The steps to perform this function can be found in Appendix B and C and are discussed in detail in Chapter IV. Following this, a simple conversion from cylindrical to polar coordinates is performed to map out the image space in Cartesian coordinates.

2. Scene Mapping

Stored images retrieved from the onboard cameras provided relative bearings and ranges to objects in the FOV. To make sense of this in the earth frame, we need to transform from the sensor frame, We simply convert from cylindrical coordinates to Cartesian coordinates and assign a GPS value to the area of each object, see Figure 11.

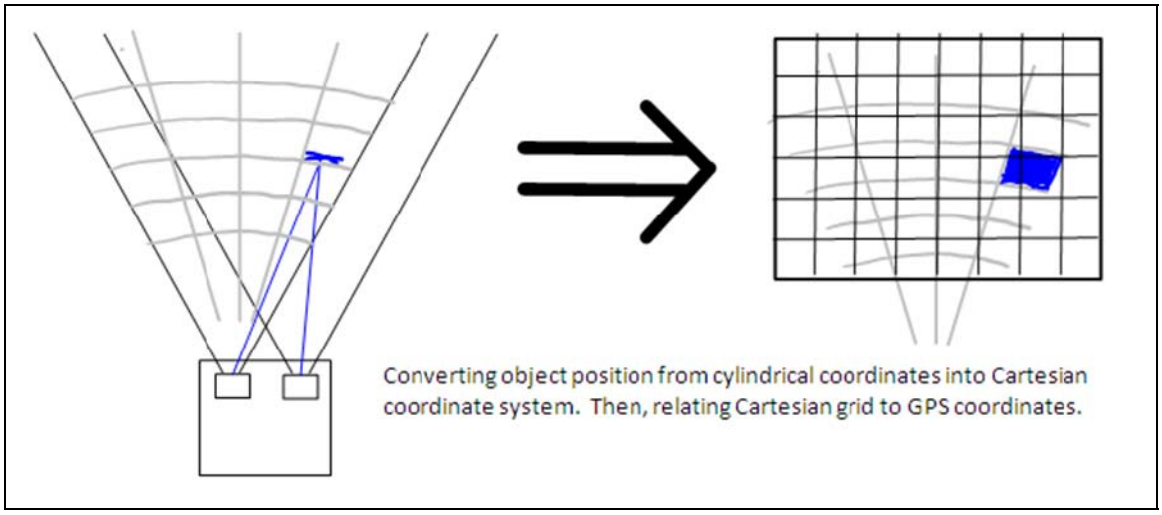


Figure 11. Converting range data to Cartesian coordinates

The conversion from cylindrical to Cartesian coordinates is simply done by the following equations:

$$x = r\cos(\theta) \quad (1.9)$$

$$y = r\sin(\theta) \quad (1.10)$$

$$z = z \quad (1.11)$$

To simplify the problem of mapping a curved coordinate plane to a square grid pattern we assume that all objects are flat and fall on the plane that is exactly r distance away from the sensor. This is a good assumption for objects that fill only small angles in the FOV or are far away. Since, we expect the object identified in this application to be both relatively close and fill a significant portion of the FOV, the mapped scene will contain range error. That range error will mostly be a function of object thickness that cannot be measured directly using the ranging techniques described within this thesis. However, multiple measurements on the same object from different angles may provide the additional information to determine object depth as well as range.

Finally, the x, y, z coordinates are referenced to the current position of the robot platform to give GPS coordinates of the object. The code used to perform this conversion is provided in Appendix D.

THIS PAGE INTENTIONALLY LEFT BLANK

III. EXPERIMENTAL DESIGN

The implementation of this concept was done such that it could be easily incorporated into the existing architecture of the NPS Applied Physics robotics lab. After careful evaluation, it was determined that the constraint this requirement imposed was that it be compatible with the onboard Rabbit BL2600 processor.

A. SENSOR

The primary sensor for this application is a commercially available camera module that is capable of RS-232 communication through a UART interface. It is a standalone device that is designed to be easily implemented in electronic designs. However, to better suit the particular requirements of this project, the manufacturer's installed lens was replaced with one that had an adjustable FOV and an interface board was constructed due to the voltage differences of the Rabbit processor and camera module.

1. Camera

The primary sensor used for this evaluation was a C328R Camera Module using an OmniVision VGA color digital CameraChip [10]. The Omnivision color digital

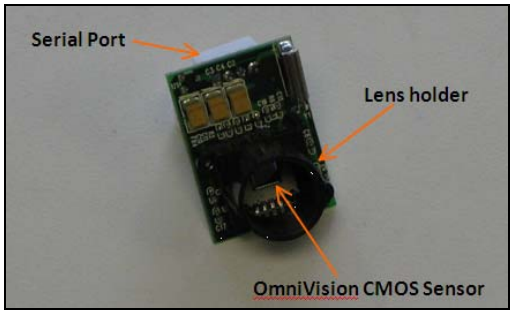


Figure 12. C328R Camera Module

Camera Chip is a CMOS sensor with on chip image processing, see Figure 12. The additional hardware adds onboard memory and converts the interface to UART for

simplified command based image processing and transfer. Sensor Options include grayscale or color images up to 16 bit and a resolution of raw images up to 160x120 pixels [11].

2. Lens

The lens was chosen to provide selectable focal lengths between 4 to 9 mm so that narrow and wide FOVs could be evaluated. This lens also provides a convenient mounting location when the internal iris is removed, as can be seen in Figure 13.

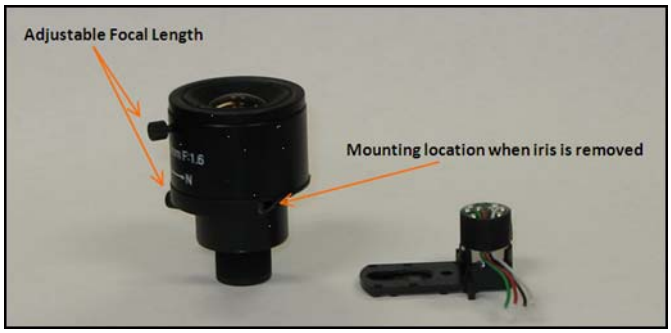


Figure 13. Variable Focal Length Lens

3. Circuit Design

One configuration issue that came up during the initial stages of sensor evaluation and testing was that of differing signal requirements for the camera and the Rabbit processor. The initial review of documentation for the C328R camera module indicated that specifications for the device were for a UART interface using an RS-232 communication protocol. The expectation was that the transmit and receive wires would be directly connected to the appropriate fittings for the desired channel to be used; however, figures in the C328R user manual indicated that the camera module actually monitored for TTL signals. The difference between the two is that one is the inverse of the other as shown in Figure 14.

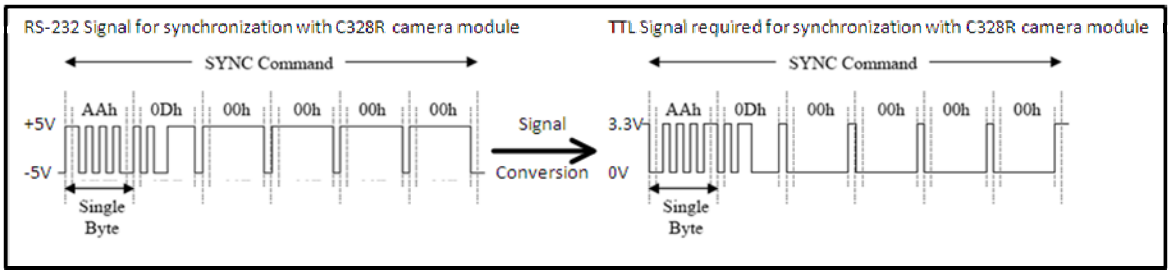


Figure 14. Communication protocol for Rabbit processor and C328R camera (from [12])

To perform this conversion, a simple Schmitt Trigger circuit was designed and integrated into the mounting platform for the sensor. The circuit effectively separates the reference voltages of the two electronic devices and inverts the signal using LM6132 Rail-to-Rail I/O Operational Amplifier. The circuit diagram is shown in Figure 15.

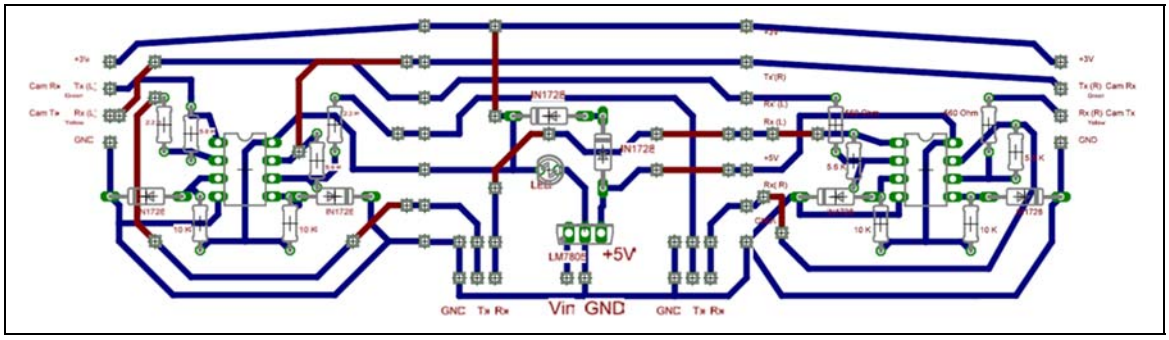


Figure 15. RS-232 to TTL Signal Conversion Circuit (See diagram in Appendix F)

B. PHYSICAL CONSTRUCTION

Construction of the test platform to evaluate this sensor configuration required a stable platform, see Figure 16, and hard mounts for fixing the relative distance (D_3) between left and right camera modules. Also, to maintain high signal quality during communication between hardware devices, the prototyped circuit boards were etched after verification that they met design parameters. This step became integral in maintaining communications at the maximum allowed rate for the camera modules at 115,200 bps.

1. Sensor Platform

Figure 16. Sensor Platform

Maintaining the relative positions of the two stereo vision sensors constant is very important if the simplified equation from 1.8 is to be implemented. Figure 16 shows how removing the lens iris provided a mounting location for the attached camera module. Also, the circuit board that was used to convert the command and data signals between the Rabbit Processor and the C328 camera module made an ideal surface mount to maintain D_3 constant.

2. Power Bus

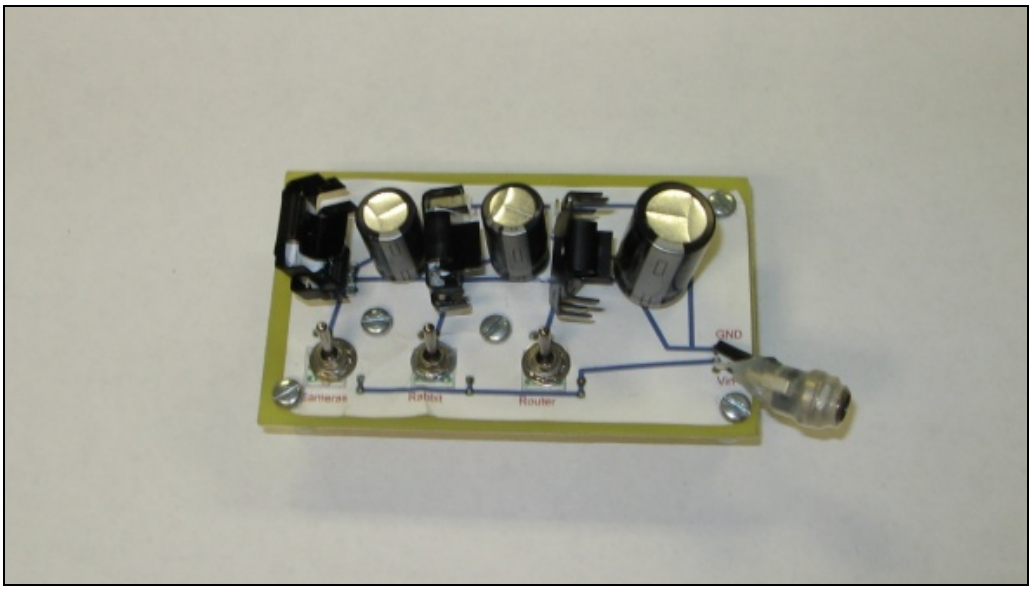
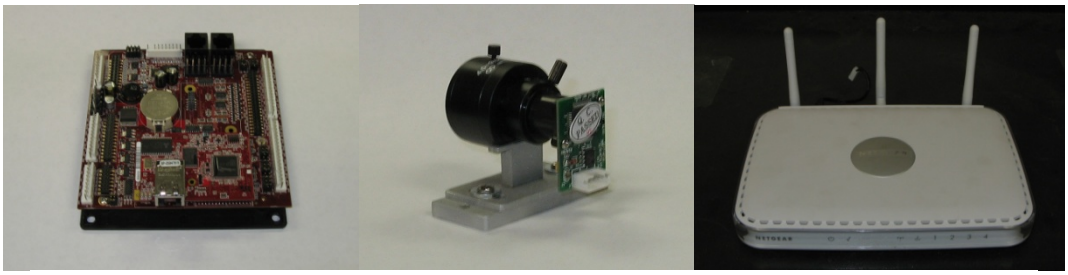


Figure 17. Power Bus

The circuitry built into the sensor platform, from Figure 15, steps voltage down from the supply voltage to power the camera at 3.3 V and the communications lines at

5V, however, the integrated sensor platform included a Rabbit processor and communications modem. This leads to the need to design a power bus. The power bus is simply a set of three 12V voltage regulators with capacitors across the output to minimize fluctuations during times of high but short current demand. It allows each component to be turned on separately during startup and, more importantly, allowed the cycling of power to individual devices if required for troubleshooting.

3. Communications Bus



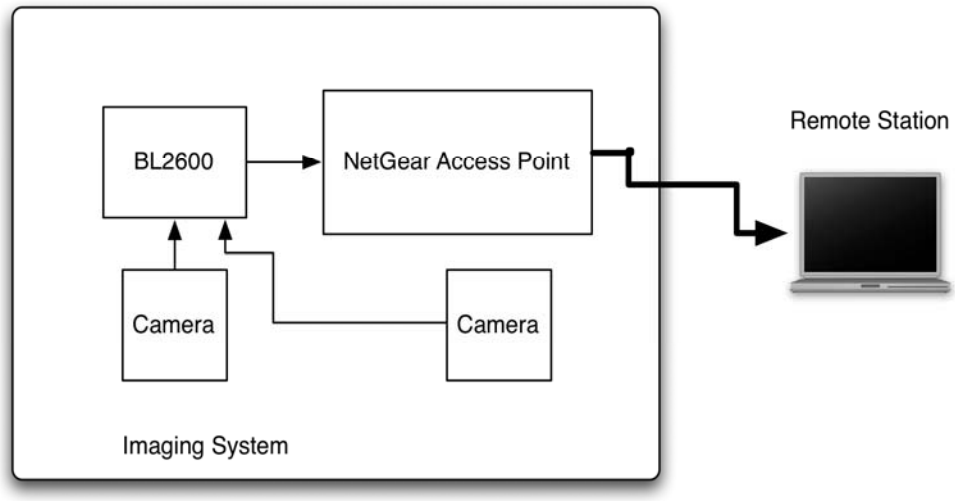


Figure 18. Communications Bus

Early in the design phase, it was apparent that the onboard processing of the Rabbit was not sufficient to handle the computations required for cross correlation of digital images, so a communications bus was added to transfer data from the camera modules to a stand-alone laptop. Based on the mobile design of the sensor assembly, the laptop could be connected wirelessly and perform the necessary computations remotely before sending commands back to the UGV control unit. For sensor implementation in

the UGV, it might become necessary to attach the laptop to the UGV chassis so it can perform all the heuristic algorithm computations on a device with greater processing speed and maintain the assignment of hardware control and monitoring functions to the Rabbit.

4. Assembly

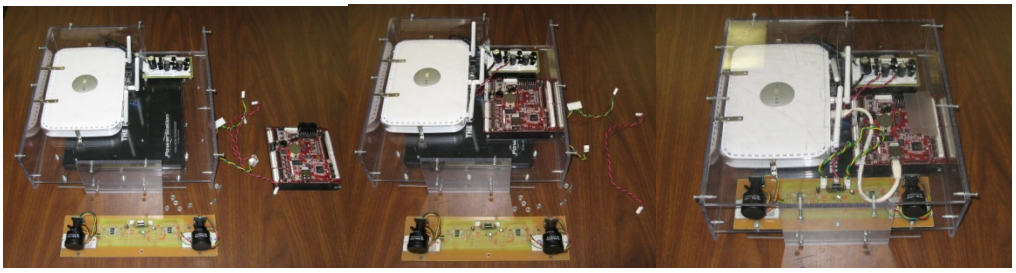


Figure 19. Sensor Housing

A plastic enclosure was constructed to protect and house the components of the stereo vision sensor unit. When implemented as a sensor on the UGV, only the sensor platform of Figure 16 is necessary but, during testing, a Rabbit processor and router were required to transfer data to the laptop. The arrangement of the components is shown in Figure 19 and assembled structure is shown in Figure 20.

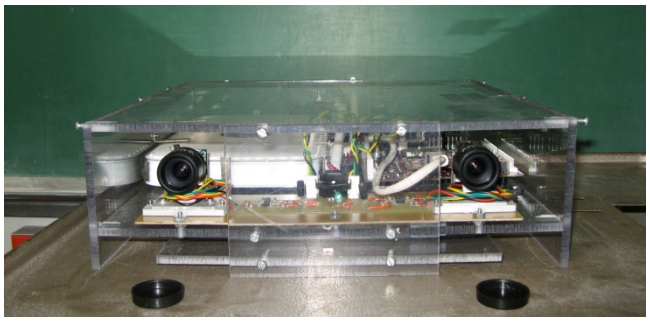


Figure 20. Stereo Vision Data Acquisition Module

IV. EXPERIMENTAL RESULTS

A. OBJECT IDENTIFICATION

Object recognition is well developed for a person who uses vision as a significant method to sense their environment. But this process is not easily quantified using lines of code and streaming data. For this, a heuristic process was implemented that first identified images in each image frame and saved those images as arrays with pixel values of either one or zero. Then the arrays defining the identified images were cross correlated to identify matches. The heuristic in this application used the MATLAB operator "contour.m,' which establishes boundaries for objects along lines of similar levels of pixel intensity. To speed up processing, each pass of the contour.m operation across the image applies a series of user generated contour maps that were found to best identify objects for the local environment. The user generated contour maps define the bounds that contour.m function follows when it creates object boundaries and were determined using trial and error for the lab environment. In any other environment, the user generated contour maps may or may not render arrays that result in correlated objects; however, the automatic generation of ideal contour mapping based on environment is not required to demonstrate the functionality of stereo vision and could be an area of future research. Finally, the portions of the image that produce range results are deleted from the image prior to subsequent passes.

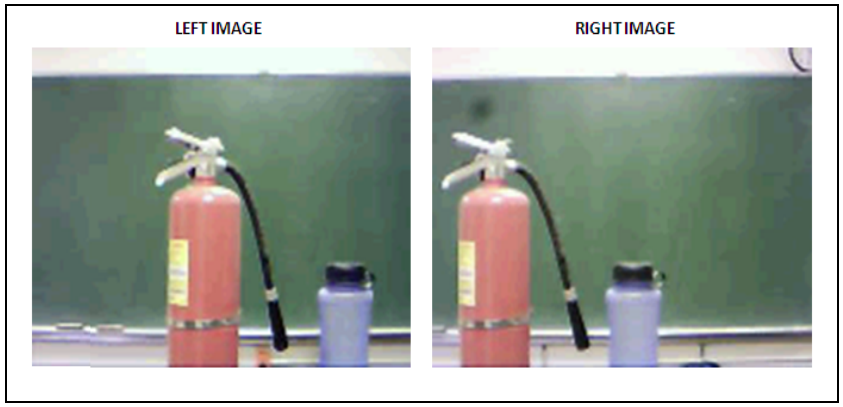


Figure 21. Example of Stereo Vision Image Capture

Figure 21 shows a raw image with two objects placed approximately 4 feet from the sensor. Figure 22 then shows how the first pass identifies a set of objects that could correlate between the two image frames.

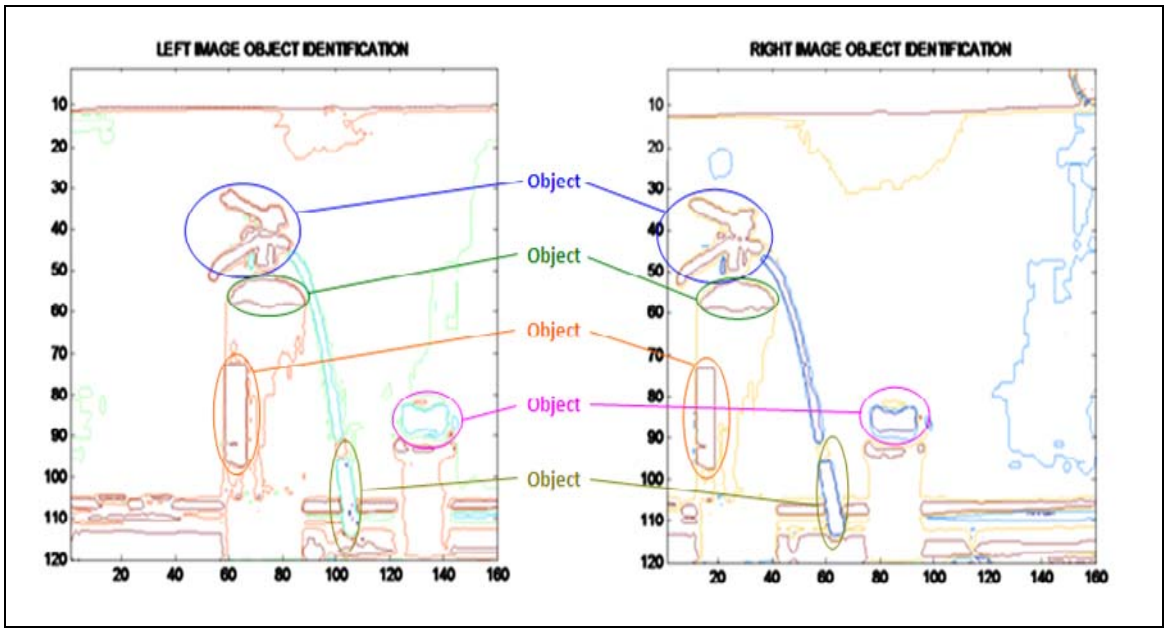


Figure 22. Object Identification using edge detection of pixel intensity level

Next, the arrays of objects for each image frame are cross correlated to identify matches. Figure 23 shows an instance where an object piece is correlated between the left and right image frames. Observe that the left image, in this case the 11th object array created using contour.m, has a 'high' correlation value with the 8th object array of the right image. The definition of 'high' correlation is subjective and was defined to be 0.7 to 1.3 of the value for self-correlation. This window for correlation values was found to provide an acceptable probability of correlation while preventing false correlation.

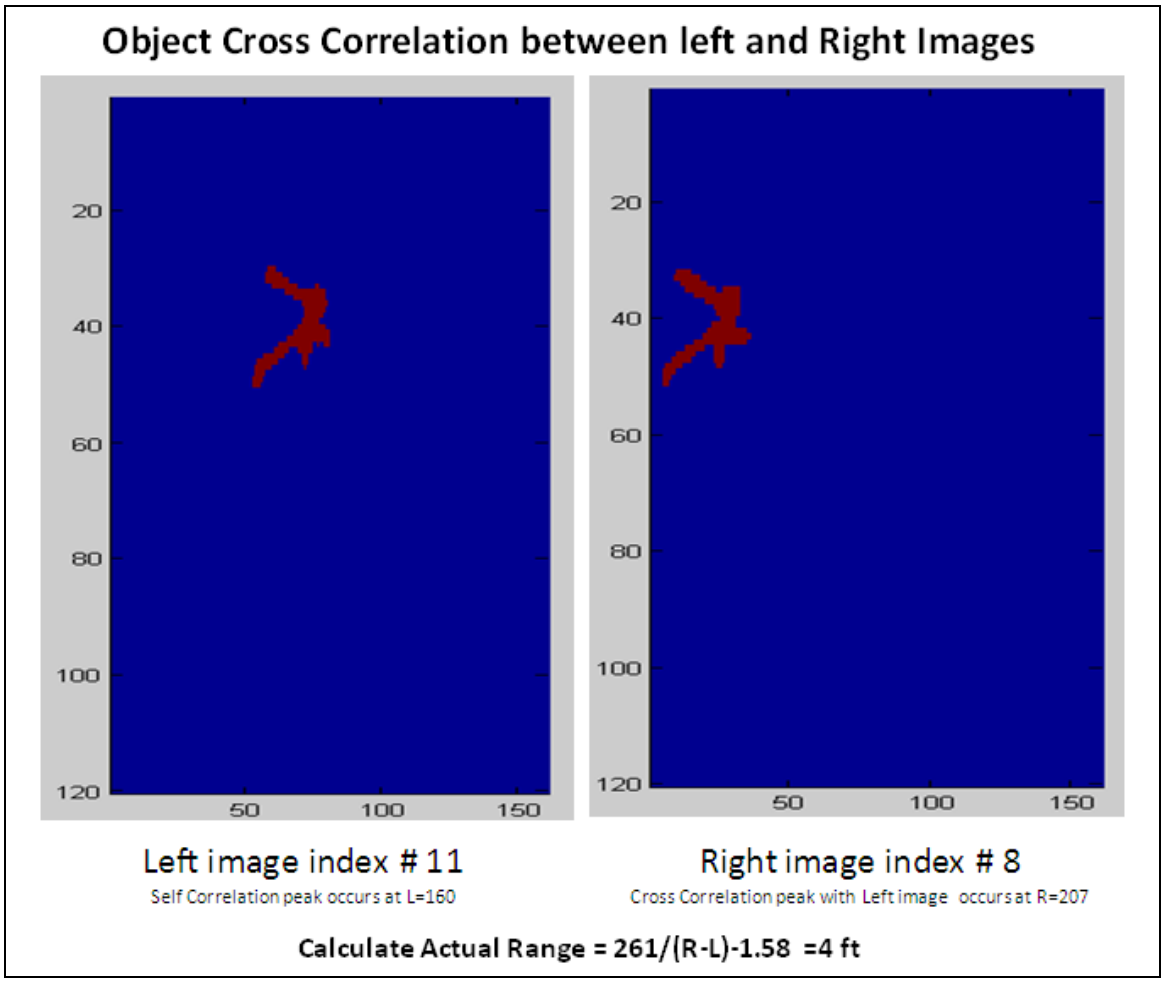


Figure 23. Object Cross Correlation

The code used to perform this operation is provided in Appendix B.

B. RANGE ACCURACY

Now that the object pieces in each image frame are correlated, it is simple to determine range. The object piece from the left image frame is self correlated and the x coordinate of the maximum value is stored as the L position of that object. Next, the object pieces are cross correlated and the x coordinate of the maximum value is stored as the R position of the object. If the constant C from Equation 1.8 is known, and the relationships of Figure 10 are applicable then range is quickly determined. Figure 23 also shows that range is calculated at the same time the images are correlated using a best fit line described in the following paragraph.

Determination of C, by estimating a best fit line, was done by performing a series of measurements on objects placed at intervals from two to thirteen feet as shown in Figure 24. The code used to perform these measurements is included as Appendix G. The graph of Figure 24 shows that a best fit line for the data points has a slope of 293 Feet per inverse pixel separation and an intercept of -0.7 Feet.

Object Range (Feet)	L-R Value (Pixels)	$\frac{1}{L\text{-}R}$ Value
2	118	0.008475
2.25	104	0.009615
2.5	99	0.010101
2.75	89	0.011236
3	84	0.011905
3.25	76	0.013158
3.5	70	0.014286
3.75	67	0.014925
4	63	0.015873
4	59	0.016949
4.25	57	0.017544
4.5	55	0.018182
4.75	52	0.019231
5	50	0.02
5.5	46	0.021739
6	44	0.022727
6.5	40	0.025
7	38	0.026316

Object Range (Feet)	L-R Value (Pixels)	$\frac{1}{L\text{-}R}$ Value
7.5	34	0.029412
8	34	0.029412
8.5	32	0.03125
8.75	31	0.032258
9	30	0.033333
9.125	30	0.033333
9.25	29	0.034483
9.375	29	0.034483
9.5	29	0.034483
9.625	28	0.035714
9.75	28	0.035714
9.875	28	0.035714
10	27	0.037037
10.5	26	0.038462
11	25	0.04
11.5	25	0.04
12	23	0.043478
13	22	0.45455

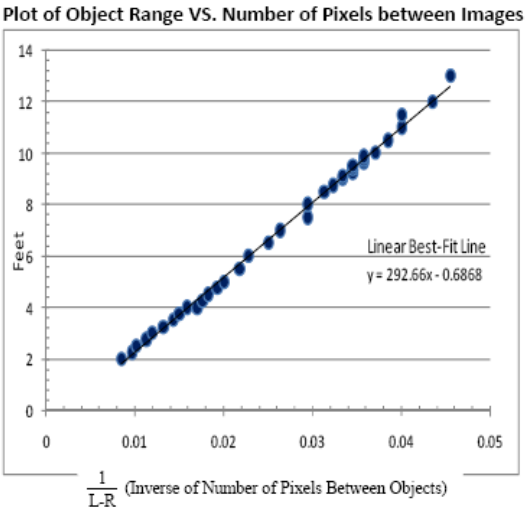


Figure 24. Table of Range Calculations for Objects with Linear Best-Fit Line

Figure 25 is another plot of the data points from Figure 24. In this graph, a quantized-best-fit line takes into account finite pixel ratios and replaces the linear best fit line. If you zoom in on the plot at ranges near 9 feet, Figure 26, you can clearly observe that the quantized-best-fit line accounts for the range deviation that occurred with the linear-best-fit equation.

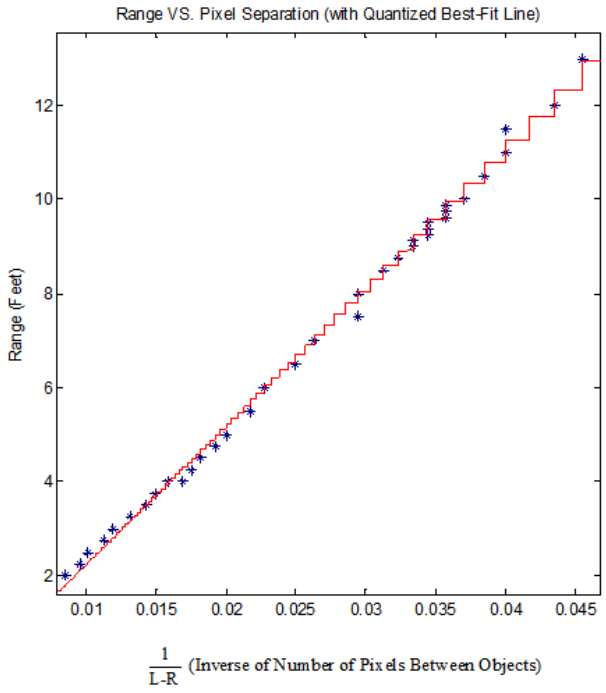


Figure 25. Range VS. Pixel Separation (with Quantized Best-Fit Line)

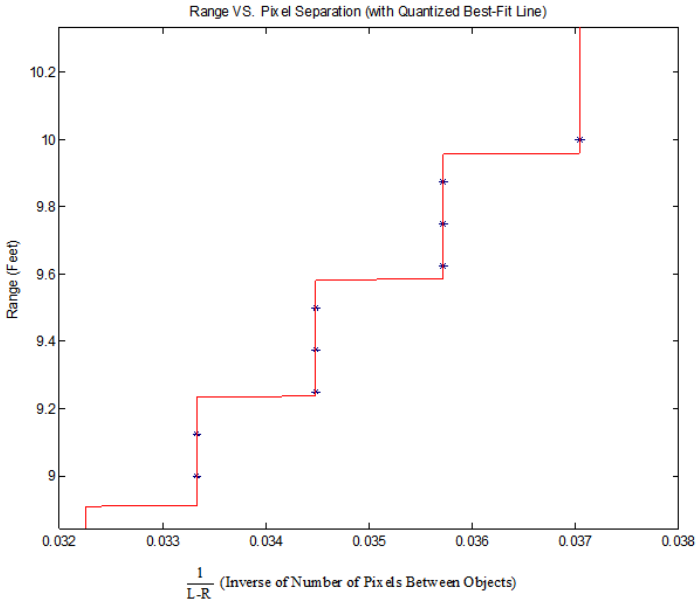


Figure 26. Range VS. Pixel Separation (zoom in on Figure 25)

When quantization is introduced, a range window develops for each value of pixel separation. As can be observed in Figure 25 and 26, that window grows in size as the separation between pixels is reduced. Therefore, there is increased error in range

calculations the further away from the object that the UGV is. Acceptable range error is subjective; however, by extrapolating Figure 25, range can be determined up to about 40 feet with 80% accuracy.

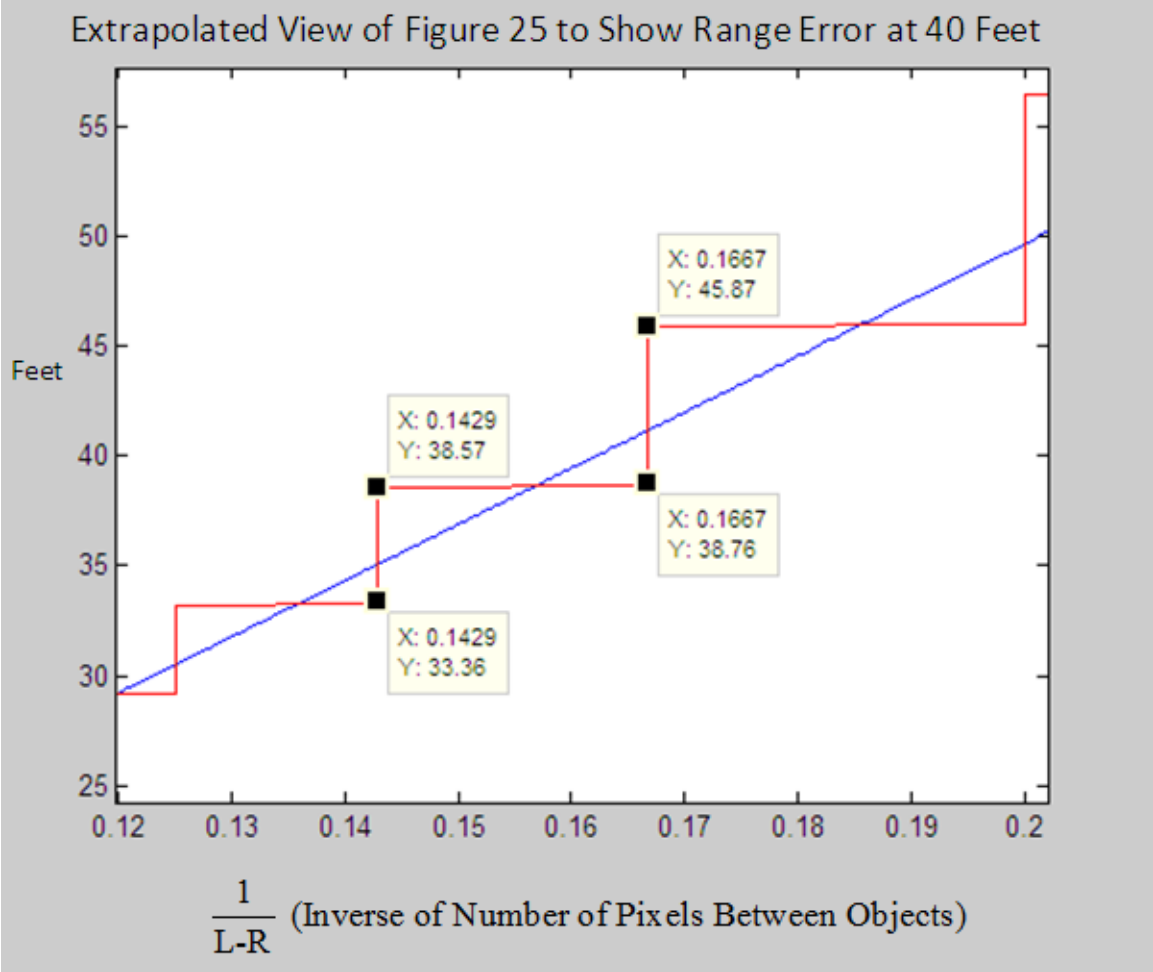


Figure 27. Object Range Determination Outside of Laboratory

Figure 27 shows that an object that is physically at a range of 38.76 feet, exactly on the line between pixel shift points, could have an error in pixel shift measurement (L-R) of one unit that causes it to register either 45.87 or 33.42 feet. In either case, the measured range is within 20% of actual.

C. SCENE MAPPING

Mapping the objects that were identified in the scene and correlated for range is a fairly simple process. Figure 27 shows how the equations of Chapter II paragraph B.2 were implemented in the Matlab code from Appendix D to map the images of the fire extinguisher, water bottle, and rear wall of the room as pictured.

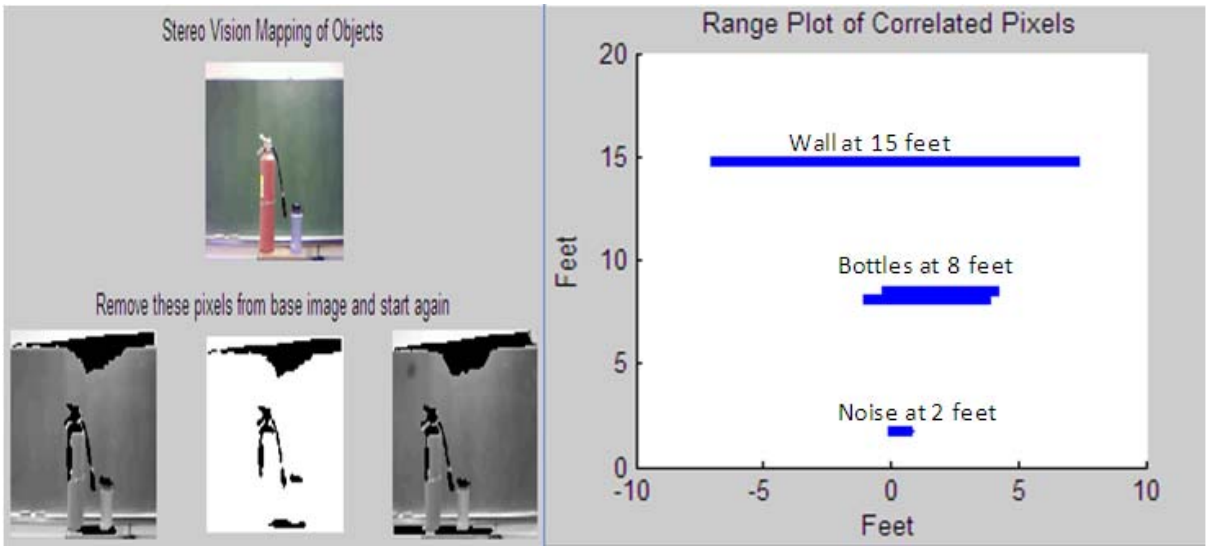


Figure 28. Scene Mapping Results

The sensor was placed 15 feet from the wall with the fire extinguisher and water bottle on a platform approximately 8 feet away.

The simple scene mapping algorithm created for this project was successful; however, there are observable errors that occur such as the object that is identified in Figure 27 to be at approximately 2 feet from the sensor. Similarly, there is a distant object, not shown on the figure that would be placed an additional 25 feet beyond the wall. Removing these measurement errors from a stored database could be done by comparing the history of measurements and maps created as the UGV navigates a space and ignoring any object that does not periodically map to the same position within the error of measurement, as shown in Figure 25.

THIS PAGE INTENTIONALLY LEFT BLANK

V. FUTURE WORK AND CONCLUSIONS

A. FUTURE WORK

The stereo vision system developed in this report is not very mature and could benefit from an assortment of future research or upgrades. They include things such as upgraded sensors, improved heuristics for object identification, and the development of a database for storage and recovery object information. These areas of future work have the greatest potential of moving the sensor that was successfully developed and tested in the laboratory to one that could be successfully employed on a UGV.

1. Implementation of Better Cameras

A sensor that was purchased, but not implemented in this design, was a 1.3 Megapixel Micron Camera on Chip device. It would attach to the lens mounts exactly in the same way. The main difference in this upgraded camera is not that it has better pixel resolution but that it transmits data on 8 channels, so image transmission occurs much more quickly. The downside of this upgrade is that it will take up more ports on the Rabbit unless an interface is created to change its output to serial but then, if that happens, it becomes just like the C328R.

2. Develop Heuristic Hierarchy for Dynamic Environments

When the sensor was moved from the lab to alternate environments, the results that were obtained in the controlled environment of the lab were not readily obtained without manipulation of the user defined contour mapping array. However, once a contour mapping array was found that worked to identify an object in its environment, it continued to work. For that reason, it would be useful to develop a catalog of user defined contour mapping arrays that are intelligently chosen based on the sensed environment.

3. Database Storage and Recall

Ultimately, the goal of an autonomous UGV is to operate and interact with its environment. For that to happen, it must be capable of recognizing objects that leave the

immediate area and be able to return later to reacquire the same object. For that to happen, the UGV requires a database for the storage of object information and characteristics.

B. CONCLUSION

The tools to record images using two cameras for stereo vision were successfully created with results following what was predicted in Figure 25, where range is a function of pixel separation between peaks in the cross-correlation of object arrays. Similarly, the heuristic process that employed user defined contour maps for object identification was sufficient to pick out objects from a controlled lab experiment in which the test objects are uniform in color and contrast significantly from the background.

Acceptable range error for a stereo vision system is a function of user preference but it was shown that the range error increases at an increasing rate as object are moved farther away. To improve range error, the user could pick a sensor with a higher pixel count because the calculated range was dependent on the pixel shift (L-R). A sensor with more pixels would reduce the error for an object at a given range.

When the sensor was moved from the controlled environment of the lab to an environment that contained more objects and color textures, such as an open courtyard, the number of false detects and failure to detect on objects increased. Similarly, as the complexity of the images increased resulting in the number of objects being compared using cross-correlation increased, processing the images took more time. Although these problems did not contradict the observations made in the lab, it was apparent that a more developed system to identify objects is required for autonomous operations.

1. Recommendations Prior to Utilization on UGV

Despite the setbacks observed as image complexity increases, the C328 camera module performed as predicted and could be used on a UGV for scene mapping in simple environments. The sensor is small, lightweight, and fully compatible with the Rabbit processor used on NPS Physics Department Robotics Lab platforms. Prior to implementation, it is recommended that improvements in object identification and the

development of an object database for storing and recalling the position of objects as they are identified be made in order for the sensor to be effective. Similarly, the object database should be called on by the processor to aid in the selection of contour maps that are likely to identify objects predicted to be in the field of view.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A – MASTER MATLAB CODE

```
% Keith Baravik
% Naval Post Graduate School Thesis 2009
% Main Program file for Stereo Vision

%% Establish Global variables
clear all
NUM_PIXEL_THRESHOLD =20; % # points required to be considered as
possible object
% R-G-B pixel maps
PR1(1:120,1:160)=0;PR2(1:120,1:160)=0;PR3(1:120,1:160)=0;
PL1(1:120,1:160)=0;PL2(1:120,1:160)=0;PL3(1:120,1:160)=0;
% Greyscale pixel map

%% Gather Digital Data from Camera (or import files for testing)
IMPORT_DATA_FILES
% convert to grayscale for test
PL1=rgb2gray(picL);
PR1=rgb2gray(picR);
%% Process image files for stereo vision
IDENTIFY_OBJECTS
COMPARE_MATRICES
%    clear RobjectR
%    clear RobjectL
%    clear RtempObj
%    clear L_OBJECT
%    clear R_OBJECT
%    clear cL
%    clear hL
%    clear cR
%    clear hR
%IDENTIFY_OBJECTS
%COMPARE_MATRICES
%    clear RobjectR
%    clear RobjectL
%    clear RtempObj
%    clear L_OBJECT
%    clear R_OBJECT
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B – OBJECT RECOGNITION MATLAB CODE

```
% Keith Baravik
% Naval Post Graduate School Thesis 2009
% Identify Objects in images

CONTOUR_MAP=[5 65 95 150 190 256];
subplot(1,2,1)
PL1a = medfilt2(PL1,[2 2]); %filters out noise
PL1b = medfilt2(PL1a,[2 2]); %filters out noise
PL1A = medfilt2(PL1b,[2 2]); %filters out noise
[cL,hL] = contourf(PL1A,CONTOUR_MAP); clabel(cL,hL), colorbar
subplot(1,2,2)
PR1a = medfilt2(PR1,[2 2]); %filters out noise
PR1b = medfilt2(PR1,[2 2]); %filters out noise
PR1A = medfilt2(PR1b,[2 2]); %filters out noise
[cR,hR] = contourf(PR1A,CONTOUR_MAP); clabel(cR,hR), colorbar
[a,LengthMat]=size(cR);

n=1;
ROBJ=1;
disp('Right Image')
while n<(LengthMat-1)
    %if cR(2,n)==1
    %    n=3;
    %end
    for J=(n+1):(cR(2,n)+n)
        if isnan(cR(1,J))==1
            cR(1,J)=(cR(1,J-1)+cR(1,J+1))/2;
            cR(2,J)=(cR(2,J-1)+cR(2,J+1))/2;
        end
    end
    for J=(n+1):(cR(2,n)+n)
        RobjectR{ROBJ}(1,J-n)=cR(1,J);
        RobjectR{ROBJ}(2,J-n)=cR(2,J);
    end
    RobjectR{ROBJ}(3,1)=cR(1,n);
    RobjectR{ROBJ}(3,2)=cR(2,n);
    RobjectR{ROBJ}(3,1:2);
    n=n+cR(2,n)+1;
    ROBJ=ROBJ+1;
end

[a,LengthMat]=size(cL);
n=1;
ROBJ=1;
disp('Left Image')
while n<(LengthMat-1)
    %if cR(2,n)==1
    %    n=3;
    %end
    for J=(n+1):(cL(2,n)+n)
        if isnan(cL(1,J))==1
```

```
            cL(1,J)=(cL(1,J-1)+cL(1,J+1))/2;
            cL(2,J)=(cL(2,J-1)+cL(2,J+1))/2;
        end
    end
    for J=(n+1):(cL(2,n)+n)
        RobjectL{ROBJ}(1,J-n)=cL(1,J);
        RobjectL{ROBJ}(2,J-n)=cL(2,J);
    end
    RobjectL{ROBJ}(3,1)=cL(1,n);
    RobjectL{ROBJ}(3,2)=cL(2,n);
    RobjectL{ROBJ}(3,1:2);
    n=n+cL(2,n)+1;
    ROBJ=ROBJ+1;
end

%% Creat MAT for objects in LEFT image
figure
MAT_L(1:120,1:160)=0;
[z,Lindexmax]=size(RobjectL);
for Lindex=1:Lindexmax

L_OBJECT{Lindex}(:,:)=poly2mask(RobjectL{Lindex}(1,:),RobjectL{Lindex}(
2,:),120,160);
    MAT_L=MAT_L(:,:)+L_OBJECT{Lindex}(:,:);
end
    imagesc(MAT_L);
    drawnow;

%% Creat MAT for objects in RIGHT image
figure
MAT_R(1:120,1:160)=0;
[z,Rindexmax]=size(RobjectR);
for Rindex=1:Rindexmax

R_OBJECT{Rindex}(:,:)=poly2mask(RobjectR{Rindex}(1,:),RobjectR{Rindex}(
2,:),120,160);
    MAT_R=MAT_R(:,:)+R_OBJECT{Rindex}(:,:);
end
    imagesc(MAT_R);
    drawnow;
```

APPENDIX C – RANGE CALCUALTION MATLAB CODE

```
% Keith Baravik
% Naval Post Graduate School Thesis 2009
% Identify Objects in images

CONTOUR_MAP=[5 65 95 150 190 256];
subplot(1,2,1)
PL1a = medfilt2(PL1,[2 2]); %filters out noise
PL1b = medfilt2(PL1a,[2 2]); %filters out noise
PL1A = medfilt2(PL1b,[2 2]); %filters out noise
[cL,hL] = contourf(PL1A,CONTOUR_MAP); clabel(cL,hL), colorbar
subplot(1,2,2)
PR1a = medfilt2(PR1,[2 2]); %filters out noise
PR1b = medfilt2(PR1,[2 2]); %filters out noise
PR1A = medfilt2(PR1b,[2 2]); %filters out noise
[cR,hR] = contourf(PR1A,CONTOUR_MAP); clabel(cR,hR), colorbar
[a,LengthMat]=size(cR);

n=1;
ROBJ=1;
disp('Right Image')
while n<(LengthMat-1)
    %if cR(2,n)==1
    %    n=3;
    %end
    for J=(n+1):(cR(2,n)+n)
        if isnan(cR(1,J))==1
            cR(1,J)=(cR(1,J-1)+cR(1,J+1))/2;
            cR(2,J)=(cR(2,J-1)+cR(2,J+1))/2;
        end
    end
    for J=(n+1):(cR(2,n)+n)
        RobjectR{ROBJ}(1,J-n)=cR(1,J);
        RobjectR{ROBJ}(2,J-n)=cR(2,J);
    end
    RobjectR{ROBJ}(3,1)=cR(1,n);
    RobjectR{ROBJ}(3,2)=cR(2,n);
    RobjectR{ROBJ}(3,1:2);
    n=n+cR(2,n)+1;
    ROBJ=ROBJ+1;
end

[a,LengthMat]=size(cL);
n=1;
ROBJ=1;
disp('Left Image')
while n<(LengthMat-1)
    %if cR(2,n)==1
    %    n=3;
    %end
    for J=(n+1):(cL(2,n)+n)
        if isnan(cL(1,J))==1
```

```
            cL(1,J)=(cL(1,J-1)+cL(1,J+1))/2;
            cL(2,J)=(cL(2,J-1)+cL(2,J+1))/2;
        end
    end
    for J=(n+1):(cL(2,n)+n)
        RobjectL{ROBJ}(1,J-n)=cL(1,J);
        RobjectL{ROBJ}(2,J-n)=cL(2,J);
    end
    RobjectL{ROBJ}(3,1)=cL(1,n);
    RobjectL{ROBJ}(3,2)=cL(2,n);
    RobjectL{ROBJ}(3,1:2);
    n=n+cL(2,n)+1;
    ROBJ=ROBJ+1;
end

%% Creat MAT for objects in LEFT image
figure
MAT_L(1:120,1:160)=0;
[z,Lindexmax]=size(RobjectL);
for Lindex=1:Lindexmax

L_OBJECT{Lindex}(:,:)=poly2mask(RobjectL{Lindex}(1,:),RobjectL{Lindex}(
2,:),120,160);
    MAT_L=MAT_L(:,:)+L_OBJECT{Lindex}(:,:);
end
    imagesc(MAT_L);
    drawnow;

%% Creat MAT for objects in RIGHT image
figure
MAT_R(1:120,1:160)=0;
[z,Rindexmax]=size(RobjectR);
for Rindex=1:Rindexmax

R_OBJECT{Rindex}(:,:)=poly2mask(RobjectR{Rindex}(1,:),RobjectR{Rindex}(
2,:),120,160);
    MAT_R=MAT_R(:,:)+R_OBJECT{Rindex}(:,:);
end
    imagesc(MAT_R);
    drawnow;
% Compare matrices and Range using Cross Correlation

disp('COMPARING MATRICES')
%check ach matrix against all others from other image for correlation
CORR_NUMBER=1;
REMOVE_MAT_L(1:120,1:160)=0;
REMOVE_MAT_R(1:120,1:160)=0;
for N=1:(Lindexmax-1)
    %
    [SELF_COR,X_left_img]=max(max(xcorr2(double(L_OBJECT{N}))));
    for M=1:(Rindexmax-1)
        %[V,Xrng]=

[CROSS_COR,X_right_img]=max(max(xcorr2(double(L_OBJECT{N}),double(R_OBJ
ECT{M}))));
```

```
        if
(sum(sum(L_OBJECT{N}(:,:)))<8000)&&(sum(sum(R_OBJECT{M}(:,:)))<8000)&&(
sum(sum(L_OBJECT{N}(:,:)))>50)&&(sum(sum(R_OBJECT{M}(:,:)))>50)
        if (CROSS_COR>(SELF_COR*0.7))&&(CROSS_COR<(SELF_COR*1.3))&&
(sum(sum(L_OBJECT{N}(:,:))))>(sum(sum(R_OBJECT{M}(:,:)))*0.7)&&(sum(sum
(L_OBJECT{N}(:,:))))<(sum(sum(R_OBJECT{M}(:,:)))*1.3)
            disp('CORRELATE')
            Range=X_right_img-X_left_img;

[SELF_COR,Y_left_img]=max(max(xcorr2(double(L_OBJECT{N}'))));

[CROSS_COR,Y_right_img]=max(max(xcorr2(double(L_OBJECT{N}'),double(R_OB
JECT{M}'))));
            if Range>0
            figure %new figure

            subplot(1,3,1)
            imagesc(L_OBJECT{N})
            Z={'Left Image Index Number =' num2str(N)};
            xlabel(Z)
            subplot(1,3,3)
            imagesc(R_OBJECT{M})
            Z={'Right Image Index Number =' num2str(M)};
            xlabel(Z)
            subplot(1,3,2)
            mesh(xcorr2(double(L_OBJECT{N}),double(R_OBJECT{M})));
            Z={'Range =' num2str(Range)};
            xlabel(Z)
            title('Object Cross Correlation between left and Right
images')
            CORR_NUMBER = CORR_NUMBER+1;
            REMOVE_MAT_L=L_OBJECT{N}(:,:)+REMOVE_MAT_L(:,:);
            REMOVE_MAT_R=R_OBJECT{M}(:,:)+REMOVE_MAT_R(:,:);
            end
        end
        end
    end
end
figure
REMOVE_MAT_L2=(sign(imcomplement(REMOVE_MAT_L*256))+1)/2;
REMOVE_MAT_R2=(sign(imcomplement(REMOVE_MAT_R*256))+1)/2;
subplot(1,3,2)
imshow(REMOVE_MAT_L2);
title('Remove these pixels from base image and start again')
subplot(1,3,1)
PL1=PL1.*uint8(REMOVE_MAT_L2);
imshow(PL1)
subplot(1,3,3)
PR1=PR1.*uint8(REMOVE_MAT_R2);
imshow(PR1)
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D – SCENE MAPPING MATLAB CODE

```
%MAP_TO_GRID

L=size(Grid_Piece_Range);

subplot(3,3,2)
imshow(picL);
title('Stereo Vision Mapping of Objects');
subplot(3,3,8)
axis([-20 20 0 20]);
title('Range Plot of Correlated Pixels');
hold on
for p=1:L(2)
    for q=81:-1:2
        plot([(q-81)*Grid_Piece_width{p}(q)*(261/Grid_Piece_Range(p)-
1.58)*cosd(11)/160,(q-82)*Grid_Piece_width{p}(q-
1)*(261/Grid_Piece_Range(p)-
1.58)*cosd(11)/160],[(261/Grid_Piece_Range(p)-
1.58),(261/Grid_Piece_Range(p)-1.58)],'-b','LineWidth',4);
    end
    for q=1:81
        plot([abs(q)*Grid_Piece_width{p}(q)*(261/Grid_Piece_Range(p)-
1.58)*cosd(11)/160,abs(q+1)*Grid_Piece_width{p}(q+1)*(261/Grid_Piece_Ra
nge(p)-1.58)*cosd(11)/160],[(261/Grid_Piece_Range(p)-
1.58),(261/Grid_Piece_Range(p)-1.58)],'-b','LineWidth',4);
    end

end
subplot(3,3,5)
imshow(REMOVE_MAT_L2);
title('Remove these pixels from base image and start again')

figure
axis([-10 10 0 20]);
title('Range Plot of Correlated Pixels');
hold on
for p=1:L(2)
    for q=81:-1:2
        plot([(q-81)*Grid_Piece_width{p}(q)*(261/Grid_Piece_Range(p)-
1.58)*cosd(11)/160,(q-82)*Grid_Piece_width{p}(q-
1)*(261/Grid_Piece_Range(p)-
1.58)*cosd(11)/160],[(261/Grid_Piece_Range(p)-
1.58),(261/Grid_Piece_Range(p)-1.58)],'-b','LineWidth',4);
    end
    for q=1:81
        plot([abs(q)*Grid_Piece_width{p}(q)*(261/Grid_Piece_Range(p)-
1.58)*cosd(11)/160,abs(q+1)*Grid_Piece_width{p}(q+1)*(261/Grid_Piece_Ra
nge(p)-1.58)*cosd(11)/160],[(261/Grid_Piece_Range(p)-
1.58),(261/Grid_Piece_Range(p)-1.58)],'-b','LineWidth',4);
    end
end
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E – DYNAMIC C CODE

```
// Dynamic C Code used to transfer data from camera to laptop

#define CINBUFSIZE  4095

#define COUTBUFSIZE 15

#define FINBUFSIZE  4095

#define FOUTBUFSIZE 15

#define TCPCONFIG 1

#memmap xmem

#define PORT 23

#use "dcrtcp.lib"

tcp_Socket echosock;

 //  Located at C:\DCRABBIT_9.62\Lib\tcpip\tpc_config.lib

 //  #define _PRIMARY_STATIC_IP    "192.168.2.50"

 //  #define _PRIMARY_NETMASK      "255.255.255.0"

//   #define MY_NAMESERVER      "192.168.2.1"

//   #define MY_GATEWAY         "192.168.2.1"

//---------- VARIABLES ------------------------//

const long baud_rate = 115200L;

char buffer[2048];

char Line[12];

unsigned int cmd[3];

unsigned int rec[6];

char Picc[4000];
```

```
int status,state,a,i,j,k,recRESP;
unsigned long int T;
unsigned long int To;
//---------- MAIN PROGRAM ----------//
void main(){
brdInit(); //initialize board
sock_init();
serCopen(baud_rate);
serFopen(baud_rate);
serMode(0);
while(1) {
//--------Continuous Loop inside MAIN PROGRAM
tcp_listen(&echosock,PORT,0,0,NULL,0);
sock_wait_established(&echosock,0,NULL,&status);
sock_mode(&echosock,TCP_MODE_BINARY);
while(tcp_tick(&echosock)) {
sock_wait_input(&echosock,0,NULL,&status);
if(sock_gets(&echosock,buffer,2048))strcpy(Line,buffer);
// SYNC command received from laptop
if(strncmp(Line, "L_SYNC", 4) == 0)
{serCwrFlush(); serCrdFlush();To=MS_TIMER;
while (1){recRESP=0;
// sends SYNC command to camera
```

```
cmd[0] = 0x0DAA;cmd[1] = 0x0000;cmd[2] = 0x0000;
if(recRESP==0){serCwrite(cmd,sizeof(cmd));
T=MS_TIMER;while((MS_TIMER-T)<200);
for (i=0;i<6;i++){rec[i] = serCgetc();} // waits for start of response
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0xD){recRESP=1;}
if((MS_TIMER-To)>10000){sock_puts(&echosock,"L_SYNC FAILED\n"); break;}
}  //ACK verified
// waits for start of response
if(recRESP==1) {while((rec[0]=serCgetc()) != 0xAA);
for (i=1;i<6;i++){rec[i] = serCgetc();} // checks for SYNC command
if(rec[0]==0xAA && rec[1]==0xD && rec[2]==0x00 && rec[3]==0x00 &&
rec[4]==0x00 && rec[5]==0x00)
{cmd[0] = 0x0EAA;cmd[1] = 0x000D;cmd[2] = 0x0000; // sends ACK command
serCwrite(cmd,sizeof(cmd));
sock_puts(&echosock,"L_SYNC COMPLETE\n");
strcpy(Line,"End");
break;}}}}
if(strncmp(Line, "R_SYNC", 4) == 0)
{serFwrFlush(); serFrdFlush(); To=MS_TIMER;
while (1){recRESP=0;
cmd[0] = 0x0DAA;cmd[1] = 0x0000;cmd[2] = 0x0000; // sends SYNC command
if(recRESP==0){serFwrite(cmd,sizeof(cmd));
T=MS_TIMER;while((MS_TIMER-T)<200);
for (i=0;i<6;i++){rec[i] = serFgetc();} // waits for start of response
```

```
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0xD){recRESP=1;}

if((MS_TIMER-To)>10000){sock_puts(&echosock,"R_SYNC FAILED\n");break;}

}  //ACK verified

// waits for start of response

if(recRESP==1) {while((rec[0]=serFgetc()) != 0xAA);

for (i=1;i<6;i++){rec[i] = serFgetc();} // checks for SYNC command

if(rec[0]==0xAA && rec[1]==0xD && rec[2]==0x00 && rec[3]==0x00 && rec[4]==0x00 && rec[5]==0x00)

{cmd[0] = 0x0EAA;cmd[1] = 0x000D;cmd[2] = 0x0000; // sends ACK command

serFwrite(cmd,sizeof(cmd));

sock_puts(&echosock,"R_SYNC COMPLETE\n");

strcpy(Line,"End");

break;}}}}

// CHANGE BAUD RATE

if(strncmp(Line, "L_BAUD", 6) == 0)

{serCwrFlush(); serCrdFlush(); To=MS_TIMER;

while (1){

if(strncmp(Line, "L_BAUD0", 7) == 0){cmd[0] = 0x07AA;cmd[1] = 0x010f;cmd[2] = 0x0000;}

if(strncmp(Line, "L_BAUD1", 7) == 0){cmd[0] = 0x07AA;cmd[1] = 0x011f;cmd[2] = 0x0000;}

serCwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serCgetc()) != 0xAA); // waits for start of response

for (i=1;i<6;i++){rec[i] = serCgetc();}
```

```
if(rec[0]==0xAA && rec[1]==0xE){sock_puts(&echosock,"L_Baud rate
115200\n");strcpy(Line,"End");break;}  //ACK verified
if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_Baud rate FAILED\n");break;}}}
if(strncmp(Line, "R_BAUD", 6) == 0)
{serFwrFlush(); serFrdFlush();To=MS_TIMER;
while (1){
if(strncmp(Line, "R_BAUD0", 7) == 0){cmd[0] = 0x07AA;cmd[1] = 0x010f;cmd[2]
=0x0000;}
if(strncmp(Line, "R_BAUD1", 7) == 0){cmd[0] = 0x07AA;cmd[1] = 0x011f;cmd[2] =
0x0000;}
serFwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);
while((rec[0]=serFgetc()) != 0xAA); // waits for start of response
for (i=1;i<6;i++){rec[i] = serFgetc();}
if(rec[0]==0xAA && rec[1]==0xE){sock_puts(&echosock,"R_Baud rate
115200\n");strcpy(Line,"End");break;}  //ACK verified
if((MS_TIMER-To)>5000){sock_puts(&echosock,"R_Baud rate FAILED\n");break;}}}
// initialize Resolution and COLOR choice
if(strncmp(Line, "L_RES160x120_GRAYSCALE", 22) == 0)
{serCwrFlush(); serCrdFlush();To=MS_TIMER;
while(1){
cmd[0] = 0x01AA;cmd[1] = 0x0300;cmd[2] = 0x0303;
serCwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);
while((rec[0]=serCgetc()) != 0xAA); // waits for start of response
for (i=1;i<6;i++){rec[i] = serCgetc();}
```

```
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x1){sock_puts(&echosock,"L_Resolution 160x120\n");strcpy(Line,"End");break;}  //ACK verified

if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RES160x120 FAILED\n");break;}}}

if(strncmp(Line, "R_RES160x120_GRAYSCALE", 22) == 0)

{serFwrFlush(); serFrdFlush();To=MS_TIMER;

while(1){

cmd[0] = 0x01AA;cmd[1] = 0x0300;cmd[2] = 0x0303;

serFwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serFgetc()) != 0xAA); // waits for start of response

for (i=1;i<6;i++){rec[i] = serFgetc();}

if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x1){sock_puts(&echosock,"R_Resolution 160x120\n");strcpy(Line,"End");break;}  //ACK verified

if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RES160x120 FAILED\n");break;}}}

if(strncmp(Line, "L_RES160x120_COLOR", 18) == 0)

{serCwrFlush(); serCrdFlush();To=MS_TIMER;

while(1){

cmd[0] = 0x01AA;cmd[1] = 0x0600;cmd[2] = 0x0303;

serCwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serCgetc()) != 0xAA); // waits for start of response

for (i=1;i<6;i++){rec[i] = serCgetc();}
```

```
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x1){sock_puts(&echosock,"L_Resolution 160x120\n");strcpy(Line,"End");break;}  //ACK verified
if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RES160x120 FAILED\n");break;}}}
if(strncmp(Line, "R_RES160x120_COLOR", 18) == 0)
{serFwrFlush(); serFrdFlush();To=MS_TIMER;
while(1){
cmd[0] = 0x01AA;cmd[1] = 0x0600;cmd[2] = 0x0303;
serFwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);
while((rec[0]=serFgetc()) != 0xAA); // waits for start of response
for (i=1;i<6;i++){rec[i] = serFgetc();}
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x1){sock_puts(&echosock,"R_Resolution 160x120\n");strcpy(Line,"End");break;}  //ACK verified
if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RES160x120 FAILED\n");break;}}}
//Set Snapshot style & acquire data (uncompressed)
if(strncmp(Line, "L_UNCOMPRESSED", 12) == 0)
{serCwrFlush(); serCrdFlush(); To=MS_TIMER;
while(1){
cmd[0] = 0x05AA;cmd[1] = 0x0001;cmd[2] = 0x0000;
serCwrite(cmd,sizeof(cmd)); T=MS_TIMER;while((MS_TIMER-T)<200);
while((rec[0]=serCgetc()) != 0xAA); // waits for start of response
for (i=1;i<6;i++){rec[i] = serCgetc();}
```

```
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x5 && rec[4]==0x0 && rec[5]==0x0){sock_puts(&echosock,"L_Uncompressed Snapshot Mode\n");strcpy(Line,"End");break;}  //ACK verified

if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_UNCOMPRESSED FAILED\n");break;}}}

if(strncmp(Line, "R_UNCOMPRESSED", 12) == 0)

{serFwrFlush(); serFrdFlush(); To=MS_TIMER;

while(1){

cmd[0] = 0x05AA;cmd[1] = 0x0001;cmd[2] = 0x0000;

serFwrite(cmd,sizeof(cmd)); T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serFgetc()) != 0xAA); // waits for start of response

for (i=1;i<6;i++){rec[i] = serFgetc();}

if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x5 && rec[4]==0x0 && rec[5]==0x0){sock_puts(&echosock,"R_Uncompressed Snapshot Mode\n");strcpy(Line,"End");break;}  //ACK verified

if((MS_TIMER-To)>5000){sock_puts(&echosock,"R_UNCOMPRESSED FAILED\n");break;}}}

//NOCONVERSION

//ODRER to process PICTURE and send data!!! IN GRAY

if(strncmp(Line, "L_RAWSNAPSHOT_GRAY", 18) == 0)

{serCwrFlush(); serCrdFlush(); To=MS_TIMER;

while(1){

cmd[0] = 0x04AA;cmd[1] = 0x0001;cmd[2] = 0x0000;

serCwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serCgetc()) != 0xAA);
```

```
for (i=1;i<6;i++){rec[i] = serCgetc();}

if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x4 && rec[4]==0x0 && rec[5]==0x0){

strcpy(Line,"End");

//Record and transmit data

while((rec[0]=serCgetc()) != 0xAA);

for (i=1;i<6;i++){rec[i] = serCgetc();}

if(rec[0]==0xAA && rec[1]==0xA && rec[2]==0x1){/*printf("Pixel data incoming...\n\n");*/}

j=0;k=0;

for (i=0;i<5;i++)

{a=0;

for(k=0;k<3840;k++)

{while((serCpeek()==-1)){if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RAWSNAPSHOT FAILED TO TRANSMIT\n");break;}}

Picc[k]=serCgetc(); /*printf("received %d or %c \n",Picc[k],Picc[k]);*/}

k=sock_write(&echosock,Picc,3840);j++;}

cmd[0] = 0x0EAA;cmd[1] = 0x000D;cmd[2] = 0x0000;

serCwrite(cmd,sizeof(cmd));

sock_puts(&echosock,"L_File Done!\n");break;}}}

if(strncmp(Line, "R_RAWSNAPSHOT_GRAY", 18) == 0)

{serFwrFlush(); serFrdFlush(); To=MS_TIMER;

while(1){

cmd[0] = 0x04AA;cmd[1] = 0x0001;cmd[2] = 0x0000;
```

```
serFwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serFgetc()) != 0xAA);

for (i=1;i<6;i++){rec[i] = serFgetc();}

if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x4 && rec[4]==0x0 && rec[5]==0x0){

strcpy(Line,"End");

//Record and transmit data

while((rec[0]=serFgetc()) != 0xAA);

for (i=1;i<6;i++){rec[i] = serFgetc();}

if(rec[0]==0xAA && rec[1]==0xA && rec[2]==0x1){/*printf("Pixel data incoming...\n\n");*/}

j=0;k=0;

for (i=0;i<5;i++)

{a=0;

for(k=0;k<3840;k++)

{while((serFpeek()==-1)){if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RAWSNAPSHOT FAILED TO TRANSMIT\n");break;}}

Picc[k]=serFgetc(); /*printf("received %d or %c \n",Picc[k],Picc[k]);*/}

k=sock_write(&echosock,Picc,3840);j++;}

cmd[0] = 0x0EAA;cmd[1] = 0x000D;cmd[2] = 0x0000;

serFwrite(cmd,sizeof(cmd));

sock_puts(&echosock,"R_File Done!\n");break;}}}

//ODRER to process PICTURE and send data!!! IN COLOR

if(strncmp(Line, "L_RAWSNAPSHOT_COLOR", 19) == 0)
```

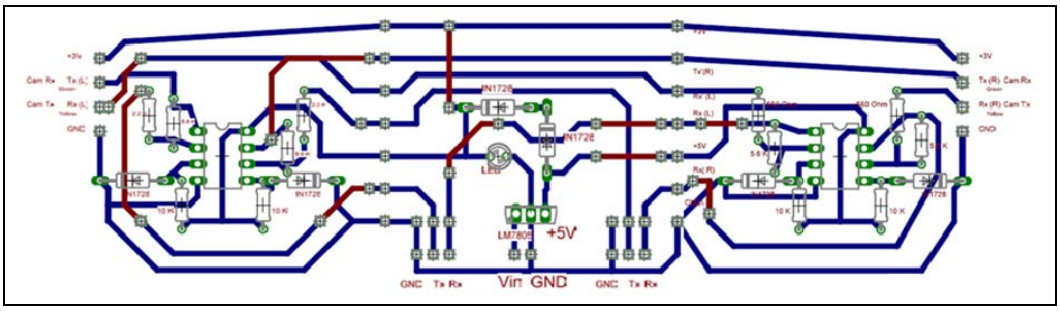
```
{serCwrFlush(); serCrdFlush(); To=MS_TIMER;
while(1){
cmd[0] = 0x04AA;cmd[1] = 0x0001;cmd[2] = 0x0000;
serCwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);
while((rec[0]=serCgetc()) != 0xAA);
for (i=1;i<6;i++){rec[i] = serCgetc();}
if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x4 && rec[4]==0x0 && rec[5]==0x0){
strcpy(Line,"End");
//Record and transmit data
while((rec[0]=serCgetc()) != 0xAA);
for (i=1;i<6;i++){rec[i] = serCgetc();}
if(rec[0]==0xAA && rec[1]==0xA && rec[2]==0x1){/*printf("Pixel data incoming...\n\n");*/}
j=0;k=0;
for (i=0;i<10;i++)
{a=0;
for(k=0;k<3840;k++)
{while((serCpeek()==-1)){if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RAWSNAPSHOT FAILED TO TRANSMIT\n");break;}}
Picc[k]=serCgetc(); /*printf("received %d or %c \n",Picc[k],Picc[k]);*/}
k=sock_write(&echosock,Picc,3840);
j++;}
cmd[0] = 0x0EAA;cmd[1] = 0x000D;cmd[2] = 0x0000;
```

```
serCwrite(cmd,sizeof(cmd));

sock_puts(&echosock,"L_File Done!\n");break;}}}

if(strncmp(Line, "R_RAWSNAPSHOT_COLOR", 19) == 0)

{serFwrFlush(); serFrdFlush(); To=MS_TIMER;

while(1){

cmd[0] = 0x04AA;cmd[1] = 0x0001;cmd[2] = 0x0000;

serFwrite(cmd,sizeof(cmd));T=MS_TIMER;while((MS_TIMER-T)<200);

while((rec[0]=serFgetc()) != 0xAA);

for (i=1;i<6;i++){rec[i] = serFgetc();}

if(rec[0]==0xAA && rec[1]==0xE && rec[2]==0x4 && rec[4]==0x0 && rec[5]==0x0){

strcpy(Line,"End");

//Record and transmit data

while((rec[0]=serFgetc()) != 0xAA);

for (i=1;i<6;i++){rec[i] = serFgetc();}

if(rec[0]==0xAA && rec[1]==0xA && rec[2]==0x1){/*printf("Pixel data incoming...\n\n");*/}

j=0;k=0;

for (i=0;i<10;i++)

{a=0;

for(k=0;k<3840;k++)

{while((serFpeek()==-1)){if((MS_TIMER-To)>5000){sock_puts(&echosock,"L_RAWSNAPSHOT FAILED TO TRANSMIT\n");break;}}

Picc[k]=serFgetc(); /*printf("received %d or %c \n",Picc[k],Picc[k]);*/}
```

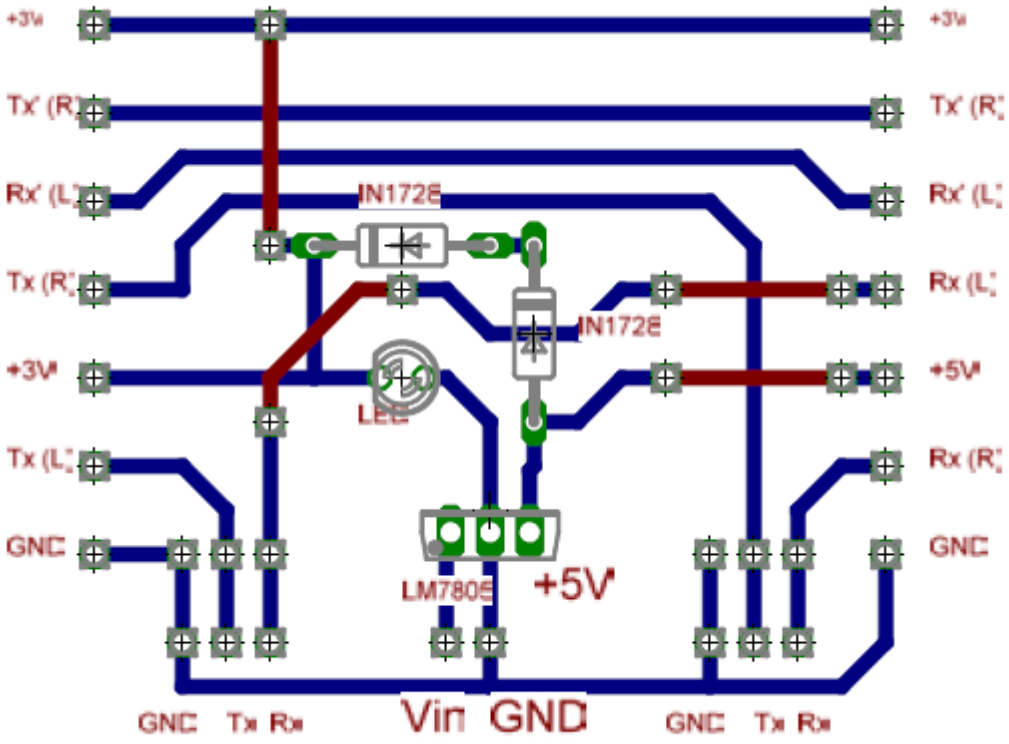
```
k=sock_write(&echosock,Picc,3840);
j++;}
cmd[0] = 0x0EAA;cmd[1] = 0x000D;cmd[2] = 0x0000;
serFwrite(cmd,sizeof(cmd));
sock_puts(&echosock,"R_File Done!\n");break;}}}
}//Closes Echosock while loop
//-----close Main & While loops
sock_err:
switch(status) {
case 1: /* foreign host closed */
break;
case -1: /* timeout */
break;}//end socket error
}//closes while loop
}//------------------ MAIN CLOSED ----------------------
```

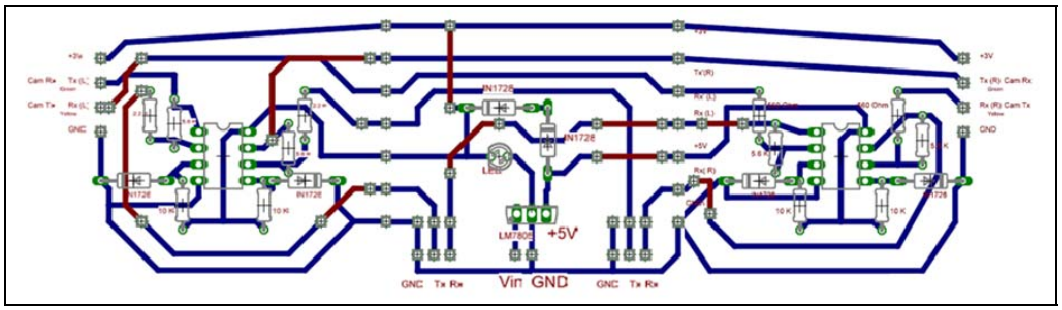
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX F – TTL SIGNAL CONVERSION CIRCUIT DIAGRAM

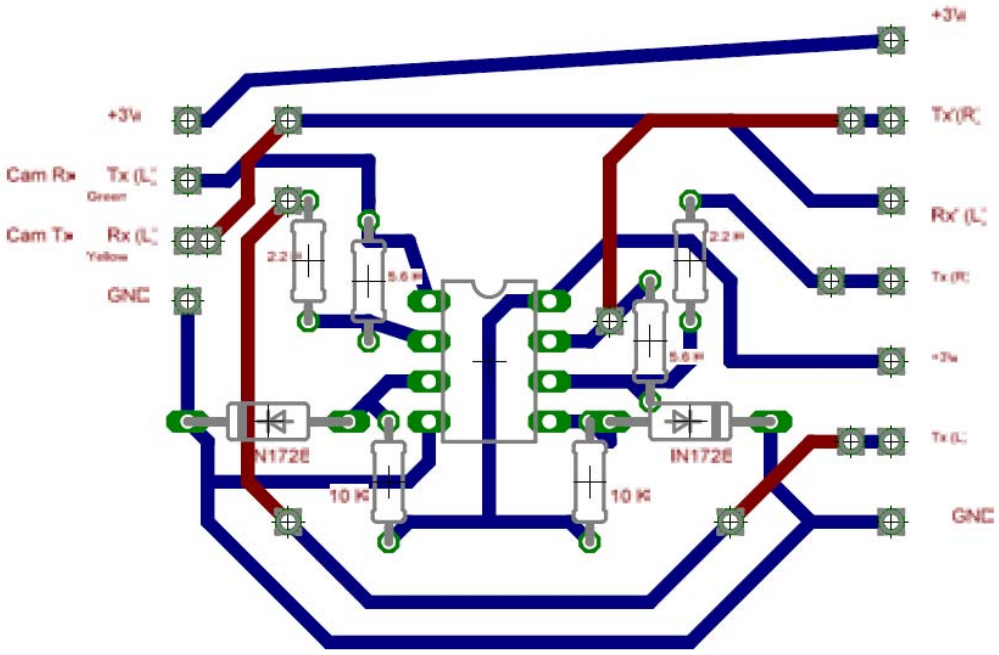


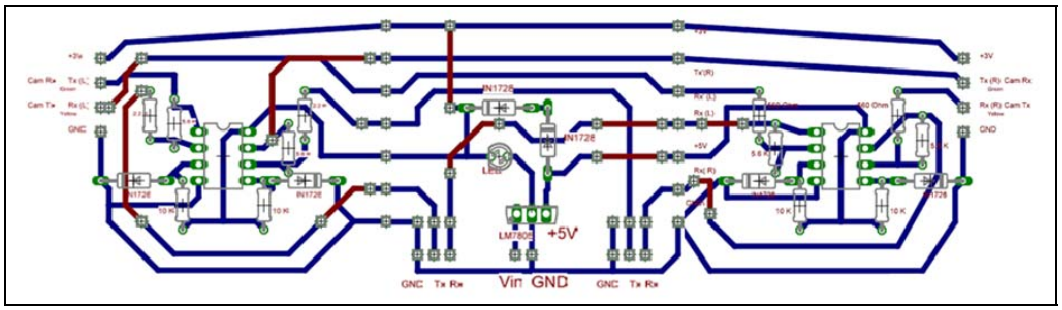
Center Region



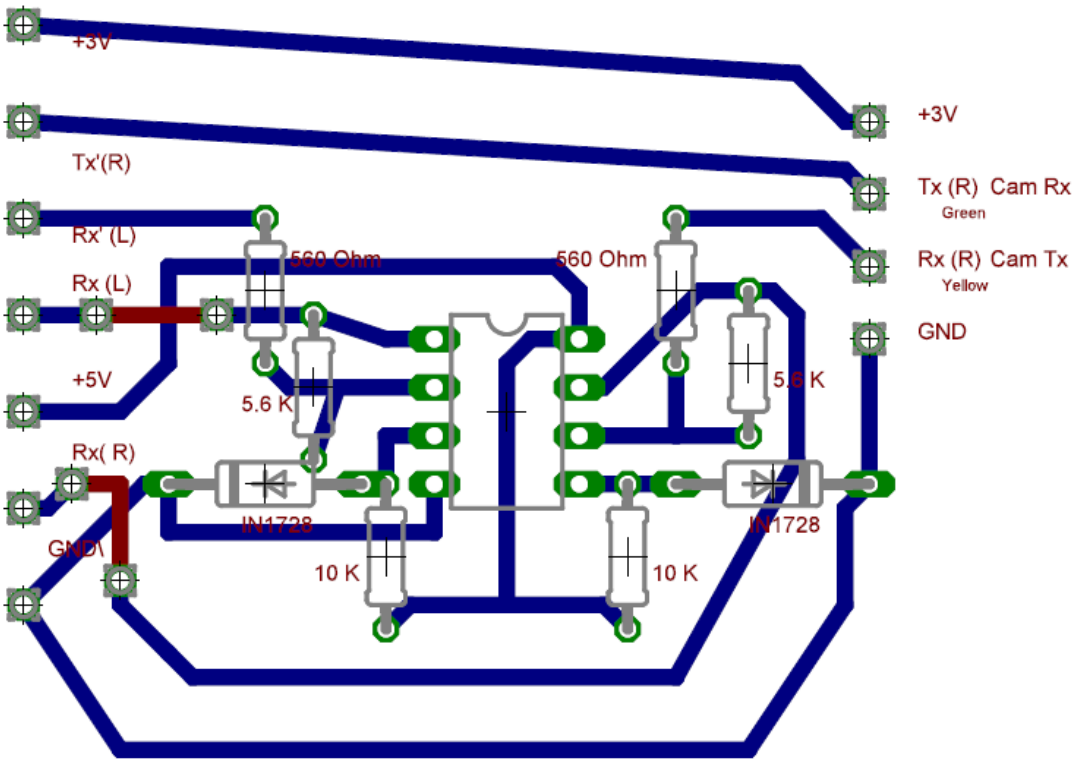

+5V
Vin GND

Left Region


+3v
Cam Rx
Tx (L)
Green
Cam Tx
Rx (L)
Yellow
GND
2.2 K
5.6 K
IN172E
10 K
Tx(R)
Rx (L)
Tx (R)
+3v
Tx (L)
GND



Right Region



THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX G – MATLAB CODE FOR GRAPHING DATA

```
%Keith Baravik
%Code for potting recorded data
% Range determination
for N=1:36
    if N==1
        Dist(N)=2;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\200_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\200_picR.png');
    elseif N==2
        Dist(N)=2.25;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\225_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\225_picR.png');
    elseif N==3
        Dist(N)=2.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\250_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\250_picR.png');
    elseif N==4
        Dist(N)=2.75;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\275_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\275_picR.png');
    elseif N==5
        Dist(N)=3;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\300_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\300_picR.png');
    elseif N==6
        Dist(N)=3.25;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\325_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\325_picR.png');
    elseif N==7
        Dist(N)=3.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\350_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\350_picR.png');
    elseif N==8
        Dist(N)=3.75;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\375_picL.png');
```

```
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\375_picR.png');
    elseif N==9
        Dist(N)=4;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\400_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\400_picR.png');
    elseif N==10
        Dist(N)=4;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\4002_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\4002_picR.png');
    elseif N==11
        Dist(N)=4.25;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\425_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\425_picR.png');
    elseif N==12
        Dist(N)=4.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\450_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\450_picR.png');
    elseif N==13
        Dist(N)=4.75;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\475_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\475_picR.png');
    elseif N==14
        Dist(N)=5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\500_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\500_picR.png');
    elseif N==15
        Dist(N)=5.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\550_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\550_picR.png');
    elseif N==16
        Dist(N)=6;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\600_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\600_picR.png');
    elseif N==17
        Dist(N)=6.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\650_picL.png');
```

```
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\650_picR.png');
    elseif N==18
        Dist(N)=7;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\700_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\700_picR.png');
    elseif N==19
        Dist(N)=7.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\750_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\750_picR.png');
    elseif N==20
        Dist(N)=8;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\800_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\800_picR.png');
    elseif N==21
        Dist(N)=8.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\850_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\850_picR.png');
    elseif N==22
        Dist(N)=8.75;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\875_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\875_picR.png');
    elseif N==23
        Dist(N)=9;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\900_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\900_picR.png');
    elseif N==24
        Dist(N)=9.125;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\912_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\912_picR.png');
    elseif N==25
        Dist(N)=9.25;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\925_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\925_picR.png');
    elseif N==26
        Dist(N)=9.375;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\937_picL.png');
```

```
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\937_picR.png');
    elseif N==27
        Dist(N)=9.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\950_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\950_picR.png');
    elseif N==28
        Dist(N)=9.625;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\962_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\962_PicR.png');
    elseif N==29
        Dist(N)=9.75;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\975_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\975_picR.png');
    elseif N==30
        Dist(N)=9.875;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\987_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\987_picR.png');
    elseif N==31
        Dist(N)=10;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1000_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1000_picR.png');
    elseif N==32
        Dist(N)=10.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1050_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1050_picR.png');
    elseif N==33
        Dist(N)=11;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1100_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1100_picR.png');
    elseif N==34
        Dist(N)=11.5;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1150_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1150_picR.png');
    elseif N==35
        Dist(N)=12;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1200_picL.png');
```

```
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1200_picR.png');
    elseif N==36
        Dist(N)=13;
        picL=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1300_picL.png');
        picR=imread('C:\Documents and Settings\ADMIN_ACCOUNT\My
Documents\THESIS\Software\Data_Folder\1300_picR.png');
    end
    %Red Ranging
figure
subplot(3,2,1)
image(picL);
subplot(3,2,2)
image(picR)

Rthresh=25;

RpicL=picL;
subplot(3,2,3)
RpicL(:,:,2)=0;
RpicL(:,:,3)=0;
for i=1:120
    for j=1:160
        if
(picL(i,j,1)<(picL(i,j,2)+Rthresh))|(picL(i,j,1)<(picL(i,j,3)+Rthresh))
            RpicL(i,j,1)=0;
        end
    end
end
imagesc(RpicL);
RpicR=picR;
subplot(3,2,4)
RpicR(:,:,2)=0;
RpicR(:,:,3)=0;
RpicR(:,:,1)=picR(:,:,1);
for i=1:120
    for j=1:160
        if
(picR(i,j,1)<(picR(i,j,2)+Rthresh))|(picR(i,j,1)<(picR(i,j,3)+Rthresh))
            RpicR(i,j,1)=0;
        end
    end
end
imagesc(RpicR);

subplot(3,2,5)
for i=1:120
    for j=1:160
        if (RpicL(i,j,1)>150)
            PRpicL(i,j)=255;
        else
            PRpicL(i,j)=0;
        end
    end
```

```
end
FPRpicL = medfilt2(PRpicL,[3 3]); %filters out noise
%Edge Detection for object ID
[B,L,Nn,A] = bwboundaries(FPRpicL);
       imagesc(FPRpicL); hold on;
       colors=['b' 'g' 'r' 'c' 'm' 'y'];
       for k=1:length(B),
         boundary = B{k};
         cidx = mod(k,length(colors))+1;
         plot(boundary(:,2), boundary(:,1),
colors(cidx),'LineWidth',2);
         %randomize text position for better visibility
         rndRow = ceil(length(boundary)/(mod(rand*k,7)+1));
         col = boundary(rndRow,2); row = boundary(rndRow,1);
         h = text(col+1, row-1, num2str(L(row,col)));
         set(h,'Color',colors(cidx),'FontSize',14,'FontWeight','bold');
       end
%

subplot(3,2,6)
for i=1:120
    for j=1:160
        if (RpicR(i,j,1)>150)
            PRpicR(i,j)=255;
        else
            PRpicR(i,j)=0;
        end
    end
end
%Edge Detection for object ID
FPRpicR = medfilt2(PRpicR,[3 3]); %filters out noise
[B,L,Nn,A] = bwboundaries(FPRpicR);
       imagesc(FPRpicR); hold on;
       colors=['b' 'g' 'r' 'c' 'm' 'y'];
       for k=1:length(B),
         boundary = B{k};
         cidx = mod(k,length(colors))+1;
         plot(boundary(:,2), boundary(:,1),
colors(cidx),'LineWidth',2);
         %randomize text position for better visibility
         rndRow = ceil(length(boundary)/(mod(rand*k,7)+1));
         col = boundary(rndRow,2); row = boundary(rndRow,1);
         h = text(col+1, row-1, num2str(L(row,col)));
         set(h,'Color',colors(cidx),'FontSize',14,'FontWeight','bold');
       end
%

%subplot(3,3,6)

%subplot(3,3,9)

%subplot(3,3,2)

L=xcorr2(PRpicL,PRpicL);
```

```
[Vl,Xl]=max(max(L));
R=xcorr2(PRpicR,PRpicR);
[Vl,Xr]=max(max(R));

Rng=xcorr2(PRpicR,PRpicL);
[V,Xrng]=max(max(Rng));

Xl
Xr
Xrng
DeltaX_L(N)=160-Xrng;
Rng=xcorr2(PRpicL,PRpicR);
[V,Xrng]=max(max(Rng));
Xrng
DeltaX_R(N)=Xrng-160;
end

figure
axes('XTick',[2 3 4 5 6 7 8 9 10 11 12 13 14 16
18],'XScale','log','YScale','log','XMinorTick','on','XMinorGrid','on');
hold on;
%plot (DistL,'b*');
xlim([2,18]);
ylim([10,160]);

% Range Simulation
%% Constants
%clear all;
T=160;
R=0;
L=160;
FOV1=36;
FOV2=36;
D3=2/3;
%% Calculations
step=1;
for R=1:160;
    A1=90-FOV1/2+FOV1*(1-L/T);
    A2=90-FOV2/2+FOV2*R/T;
    A3=180-A1-A2;
    D2=D3*sind(A2)/sind(A3);
    D1=D2*sind(A1)/sind(A3);
    RangeObj(161-R)=D2*sind(A1);
    step=step+1;
end

Measured_Dist=[2 3 4 5 6 7 8 9 10 11 12 13];
DeltaX_Possible=4:0.01:125;
figure
plot(1./DeltaX_R,Dist,'g*')
hold on
plot(1./(round(DeltaX_Possible)),292.66./DeltaX_Possible-0.6868,'r')
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] SPAWAR. SSC Technology Transfer Project. Cited: January 20, 2009. http://www.spawar.navy.mil/robots/research/Technology_Transfer/TechXfer.html.

[2] Andrew White. "UNMANNED GROUND VEHICLES – Step change in urban operations as UGVs face up to the enemy." *INTERNATIONAL DEFENCE REVIEW*. 2007, June 01.

[3] SPAWAR. VISION ROBOTICS CORPORATION. Cited: February 05, 2009. http://visionrobotics.com/vrc/index.php?option=com_content&task=view&id=72&Itemid=73.

[4] Joint Ground Robotics Enterprise. Cited: January 20, 2009. http://www.jointrobotics.com/history.php.

[5] Frank L. Pedrotti and Leno S. Pedrotti. *Introduction to Optics, Second Edition.* Upper Saddle River, New Jersey: Prentice-Hall Inc., 1993.

[6] Umesh Rajashekar. Natural Image Statistics and Human Eye Movements. *University of Texas at Austin.* Cited: January 28, 2009. http://live.ece.utexas.edu/research/eyefix/.

[7] NDT Resource Center. *Visual Acuity*. Cited: January 28, 2009. http://www.ndt-ed.org/EducationResources/CommunityCollege/PenetrantTest/Introduction/visualacuity.htm.

[8] INFOBORDER, History of Digital Cameras. Cited: February 06, 2009. http://www.infoborder.com/Digital_Camera_History/.

[9] Hasselblad. H3DII-50 to H3DII-60 Upgrade Program. Cited: February 06, 2009. http://www.hasselbladusa.com/58649.aspx.

[10] Seeed_Studio. Uart Camera Module with Jpeg compression – C328. Cited: February 23, 2009. http://www.seeedstudio.com/depot/uart-camera-module-with-jpeg-compression-c328-p-209.html.

[11] Electronics123, C328R.pdf. Cited: February 23, 2009. http://www.electronics123.net/amazon/datasheet/C328R.pdf.

[12] Electronics123, C328R_UM.pdf. Cited: February 23, 2009. http://www.electronics123.net/amazon/datasheet/C328R_UM.pdf.

[13] SPAWAR. MDARS. Cited: January 20, 2009. http://www.spawar.navy.mil/robots/land/mdars/mdars.html.

[14] Yahoo GeoCities. Light demolition carrier "Goliath." Cited: January 20, 2009. http://www.geocities.com/CapeCanaveral/Lab/1167/egoliath.html.

[15] Foster-Miller. TALON Military Robots. Cited: January 20, 2009. http://www.fostermiller.com/lemming.htm.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California